PTO 08-0614

CC=JP
DATE=19961129
KIND=A
PN=08314872

COOPERATIVE PROCESSING METHOD AMONG APPLICATION PROGRAMS

[Apurikeshon Puroguramu Kan Renkei Shori Hoho]

Toshiaki ITO, et al.

UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. NOVEMBER 2007
TRANSLATED BY: SCHREIBER TRANSLATION, INC.

PUBLICATION COUNTRY	(10):	JP
DOCUMENT NUMBER	(11):	08314872
DOCUMENT KIND	(12):	A
PUBLICATION DATE	(43):	19961129
APPLICATION NUMBER	(21):	07138641
APPLICATION DATE	(22):	19950512
INTERNATIONAL CLASSIFICATION	(51):	G06F 15/16 9/46 15/00
PRIORITY COUNTRY	(33):	N/A
PRIORITY NUMBER	(31):	N/A
PRIORITY DATE	(32):	N/A
INVENTOR(S)	(72):	Toshiaki ITO, et al.
APPLICANT(S)	(71):	Hitachi Tohoku Software Co., Ltd.
DESIGNATED CONTRACTING STATES	(81):	N/A
TITLE	(54):	COOPERATIVE PROCESSING METHOD AMONG APPLICATION PROGRAMS
FOREIGN TITLE	(54A):	Apurikeshon Puroguramu Kan Renkei Shori Hoho

/1[1]

SPECIFICATION

(54) [Title of the Invention]

Cooperative Processing Method among Application Programs

/2

[Claims]

[Claim 1] A cooperative processing method among application programs which is a cooperative processing method among plural application programs operating on plural computers connected to a computer network and

is characterized in that

an operation instruction describing utilization orders of the plural application programs and utilization procedures of respective application programs is prepared,

the operation instruction is successively transferred to the plural computers having objective application programs through the computer network,

a cooperative interface cooperating with the application programs in accordance with the utilization procedures of the application programs described in the operation instruction are provided in each of the plural application programs, when each

[1]Numbers in the margin indicate pagination in the foreign text.

cooperative interface receives the operation instruction, the cooperative interface starts up its own application programs in accordance with the utilization procedures relating to its own application programs described in the operation instruction and performs an objective control and/or giving and receiving of data, and

a series of operations based on the plural application programs are executed by patrolling computers having necessary application programs.

[Claim 2] The cooperative processing method among application programs according to Claim 1, which is characterized by comprising at least one of an all data item for indicating a summary of data items used in all application programs utilized by the operation instruction, an output data item for storing data to be given to each application program, an input data item for storing processing result of each application program, a data output procedure for indicating a procedure outputting the data stored in the output data item to the application program, a data input procedure for indicating an input procedure of data of each input data item, an input data processing procedure for processing the data inputted by the data input procedure and a

control procedure for controlling the execution of the application programs as constituents of the utilization procedures described in the operation instruction.

[Claim 3] The cooperative processing method among application programs according to Claim 1 or 2, which is characterized in that the method has a movement control means on each computer, and the movement control means ensures and manages a communication path between the cooperative interface on its own computer and other computers and controls the movement of the operation instruction according to the utilization procedures described in the operation instruction received through the communication path.

[Claim 4] The cooperative processing method among application programs according to Claim 2, which is characterized by having at least one of a start procedure for starting the application programs of the cooperative interface receiving the operation instruction, an end procedure for ending the execution of application programs, an execution procedure for executing one or more commands held by the application programs and a control script combining these procedures as control procedures that are constituents of the operation instruction.

[Claim 5] The cooperative processing method among application programs according to Claim 2 which is characterized by

having a generation procedure for generating a new operation instruction as a constituent of the operation instruction.

[Claim 6] The cooperative processing method among application programs according to Claim 1 or 2 which is characterized in that the utilization procedures are any of procedures or function names and, in the case of function name, acquire programs corresponding to the function name in the cooperative interface.

[Claim 7] The cooperative processing method among application programs according to Claim 1, 2, 3, 4, 5 or 6 which is characterized in that the utilization orders described in the operation instruction assigns a set of computer names and service entities comprising application programs operating on the computers and cooperative interfaces according to the orders of utilization of the service entities and can omit the computer names.

[Detailed Description of the Invention]

[0001]

[Field of Industrial Application] The present invention relates to a cooperative processing method among application programs which is involved in a dispersion processing system wherein a user performs processing by utilizing plural application

programs while successively combining the programs and, as necessary, can cooperate arrangements and processing among various application programs utilized by operators in performing business constructed from a series of advancing operations while cooperating the mutual operations by exchanging processing courses, processing results and processing contents with operators performing other operations and further easily change cooperative procedures of processing among individual application programs according to business.

/2 ~ 3

[0002] The present invention particularly relates to a utilization technique for flexibly combining and cooperating application programs according to operations to reduce operation load of each operator and smoothen the flow of operations and a dispersion processing system for realizing such a flexible cooperative processing method of application programs by automatizing operation procedures combining plural application programs performed by the users, operations performed at the request of each operator and acquisition of the results and systematizing utilization procedures of application programs necessary for a series of operations.

[0003]

[Prior Art] A dispersion processing system is a system wherein computers used in operations to be performed by individual operators are connected by a network to exchange data needed among individual operators. Such a system are individual utilization centers wherein one by one operations are performed by the operators with the application programs, only data needed among the operators are shared, and the data shared by the application programs and utilized by the operators can be utilized.

[0004] The construction for cooperatively operating application programs on a dispersion processing system of a four cooperative processing mode (RPC (Remote Procedure Call), conversation, file transfer and message-base) has been discussed in *Nikkei Computers:* 「Non-Synchronization and Accumulation type Communication Eliminate Weak Points of Client Servers」, 159 ~ 168 (Aug. 8, 1988) and *Nikkei Computers:* 「Message-Base Middleware for Easily Realizing Non-Synchronization Cooperation of Dispersion Applications with Each Other Rapidly Increases」, 67 ~ 74 (Sep. 20, 1993).

[0005] 「RPC」 mode is a mode wherein one module of application programs is fetched and the module is executed by other computers. 「Conversation」 mode is a mode wherein data or processing requests are exchanged among plural application programs

through a real-time communication path by connecting the programs with the real-time communication path. A system wherein application programs used in operations are divided into a processing request source (client) and a processing execution source (server) to perform cooperative processing among application programs on the client side and the server side is a client-server system. In these two modes, a cooperative processing for performing processing while taking synchronization among application programs on the client-side and the server side can be realized by promptly processing a processing request from the transmission-side application programs if the request is received and returning a processing result to the transmission-side application programs in a state that the reception-side application programs always operate and wait for a processing request from the transmission-side application programs. In these modes, cooperative procedures among the application programs for performing the cooperative processing are predetermined at the time of developing the application programs, and the cooperative procedures must be described as a program.

[0006] 「File transfer」 is a mode wherein a file is transferred among the cooperated application programs and the application programs perform processing for data described in the file when the data is received. In this mode, a cooperative

processing among non-synchronous application programs can be realized by executing the application programs with the reception of file as a trigger. [Message-base] is a mode wherein a program realizing a message-base processing receives a message sent from the transmission-side application programs, sends the message to the reception-side application programs described in the message, and a cooperative processing among the application programs is realized by executing a processing procedure corresponding to the message in the reception-side application programs receiving the message. In the message-base mode, the format of message is predetermined among the application programs performing the cooperative processing and the processing of application programs is described when the message is received.

[0007] In these four cooperative processing modes for application programs, a processing procedure necessary for cooperation must be determined for each application program and described as a program to perform business constructed from a series of operations by cooperating processing among plural application programs used by individual operations.

[0008] In logic communication path control modes among programs described in Japan Kokai 02-186737, a mode wherein a cooperative processing among application programs is realized by exchanging data among application programs operating on com-

puters connected to a network. In this mode, the cooperative processing among application programs is realized based on sharing display of pictures among the application programs by exchanging data of operations (change, supplement, deletion, etc. of data) applied by users via pictures of the application programs among the same application programs through a logic communication path.

[0009] In the cooperative processing mode among these application programs, when data and messages used in the cooperative processing are sent to application programs on other computers, the data and messages can be sent to the application programs on assigned computers by assigning sending destinations of the data and messages by any one method for name of destination computers, physical positions or physical names of destination computers.

[0010] More recently, the research and development of systems for supporting operations straddling plural operators from a system of individual utilization centers based on individual

/4

operators have been prosperously carried out. About the kinds of systems supporting these operations straddling plural operators and the contents of supported operations, systems supporting cooperative operations straddling plural persons researched and developed domestically and abroad have been discussed in H.

Ishii: ⌈Trend in Technical Research of Groupware⌋, *J. Dissertation of Information Processing Society of Japan*, 30 (12) (Dec. 1989), H. Ishii: ⌈Trend in Research of Groupware Support using Computers⌋, *Computer Software*, 8 (2), 14 ~ 26 (1985), C.A. Ellis, S.J. Gibbs and G.L. Rein: "Groupware: Some Issues and Experiences", CACM 34 (1), 38 ~ 58 (Jan. 1991).

[0011] The system supporting these cooperative operations is a system supporting that each operator cooperatively performs operations of same contents by use of same application program, it displays processing results of the application programs among the operators and provides a function sharing a user interface picture added with some operations to the processing results.

[0012]

[Problem to Be Solved by the Invention] In the above prior art, the cooperative processing among application programs is performed by predetermining data and messages exchanging among some specific application programs, describing processing programs corresponding to the predetermined data and messages in respective application programs and then exchanging the predetermined data and messages among the application programs. Therefore, procedures for the cooperative processing among application programs cannot be changed. Or even they can be changed,

the processing programs in all the application programs relating to the changes must be changed. As a result, it becomes very difficult to change cooperative processing programs among the application programs corresponding to operation contents and objects.

[0013] Moreover, it is necessary to end the execution of all application programs as objects of change to change the cooperative processing procedures among the application programs. Once application programs are used so that various users perform respective operations on a dispersion processing system and end the execution of all application programs as targets of change, the difficulty increases with increasing the number of application programs as targets of change.

[0014] Furthermore, those exchanged for performing the cooperative processing among the application programs are data or messages, therefore processing procedures at the time of receiving these data and messages is determined by the reception-side application programs receiving the data and messages, and it is impossible that transmission-side application programs changes the processing procedures of reception-side application programs. In order to realize the execution of a series of processing procedures combining plural functions provided by the reception-side application programs, i.e., combinations of plural commands to be executed at a time by the reception-side

application programs, a message for executing necessary processing procedures while taking synchronization is sent to the application programs, a pertinent procedure is executed and the execution results are combined by the transmission-side application programs, or a special message for executing plural commands must be described as a program in the reception-side application programs beforehand, thus the development efficiency lowers.

[0015] Thus, it is necessary to describe the processing procedures among the application programs beforehand, therefore it is difficult to describe the processing procedures among the application programs suited to methods utilized by individual user themselves, and it is also difficult to cooperatively processed among the application programs except that cooperative processing procedures among the application programs provided by a system are utilized.

[0016] Moreover, in order to send these data or messages to the application programs, computers in which the application programs operate must be clearly assigned by a means for univocally identifying the computers, such as names or IP addresses of the computers, etc., when computers in which the application programs operate for performing some processing are changed for some reasons, a section in which computers in the cooperative

processing procedures among application programs utilizing the pertinent application program must be changed. Therefore, the prior art is deficient in flexibility for changes of computers or network, application programs constituting a dispersion processing system.

[0017] One purpose of the present invention consists in providing a cooperative processing method among application programs which enables to perform a cooperative processing among application programs suited to a method utilized by individual user themselves.

[0018] Another purpose of the present invention consists in providing a cooperative processing method among application programs which can flexibly make changes of cooperative processing procedures among application programs without lowering the developing efficiency by giving a cooperative interface capable of giving/receiving the cooperative processing procedures among application programs in addition to data and messages, processing procedures performed in reception-side application programs by receiving the data and messages and a series of processing procedures combining plural functions provided by the reception-side application programs and executing these received procedures.

[0019]

[Means for Solving the Problem] To achieve the above purposes, the present invention is a cooperative processing method among application programs operating on plural computers connected to a computer network wherein

an operation instruction describing utilization orders of the plural application programs and utilization procedures of respective application programs is prepared,

the operation instruction is successively transferred to the plural computer having objective application programs through the computer network,

a cooperative interface cooperating with the application programs in accordance with the utilization procedures of the application programs described in the operation instruction are provided in each of the plural application programs, when each cooperative interface receives the operation instruction, the cooperative interface starts its own application program in accordance with the utilization procedures relating to its own application programs described in the operation instruction and performs an objective control and/or giving and receiving of data, and

a series of operations based on the plural application programs are executed by patrolling computers having necessary application programs.

[0020] This cooperative processing method among application programs comprises at least one of an all data item for indicating a summary of data items used in all application programs utilized by the operation instruction, an output data item for storing data to be given to each application program, an input data item for storing processing result of each application program, a data output procedure for indicating a procedure outputting the data stored in the output data item to the application program, a data input procedure for indicating an input procedure of data of each input data item, an input data processing procedure for processing the data inputted by the data input procedure and a control procedure for controlling the execution of the application programs as constituents of the utilization procedures described in the operation instruction.

[0021] The present invention preferably has a moving control means on each computer, and the moving control means ensures and manages a communication path between the cooperative interface on its own computer and other computers and controls the movement of the operation instruction according to the utiliza-

tion procedures described in the operation instruction received through the communication path.

[0022] For example, the present invention has at least one of a start procedure for starting the application programs of the cooperative interface receiving the operation instruction, an end procedure for ending the execution of application programs, an execution procedure for executing one or more commands having the application programs and a control script combining these procedures as control procedures that are constituents of the operation instruction.

[0023] The present invention may has a generation procedure for generating a new operation instruction as a constituent of the operation instruction. For example, the utilization procedures are any of procedures or function names and, in the case of function names, the programs corresponding to the function names are acquired in the cooperative interface.

[0024] The utilization orders described in the operation instruction assigns a set of computer names and names of service entities comprising application programs operating on the computers and cooperative interfaces according to utilization orders of the service entities and can omit the computer names.

[0025] More specifically, the cooperative processing among application programs in the present invention is constituted by

an operating instruction (service agent) having service entities for giving cooperative interfaces executing cooperative processing procedures among application programs in individual application program bodies involved in the cooperative processing among application programs, utilization orders for indicating movement destinations among the cooperated service entities and utilization procedures of the service entities, and movement control means (routine managers) for generating, maintaining and managing a communication path on which these service entities are moved to the movement destinations described in the utilization orders having the entities.

[0026] The service entities are arranged and executed on individual computers constituting a dispersion processing system as execution units of one application program. The cooperative interfaces constructing the service entities select control procedures possessed by the service agent, input/output data procedures and procedures pertinent to current service entities from display procedures and have functions of executing and controlling the application program bodies by executing the procedures. When a program corresponding to the executing procedures is on other computers, the cooperative interfaces have functions of copying the program on the computers operated by

the service entities and executing procedures required by the service entities.

[0027] The service entities for realizing the cooperative processing among application programs have utilization orders for describing orders cooperating the application programs in the cooperative processing as utilization orders of the service entities having pertinent application programs, and utilization procedures which comprises a control procedure for utilizing functions held by the application programs used in respective service entities, an input/output data processing procedure for transferring data items storing data necessary for actually performing the processing in application programs held by the service entities and data stored in the data items to application programs by the control procedures and storing the processing

/6

result data in the pertinent data items, and a display procedure for displaying data held by the service agent through graphical user interfaces.

[0028] The routine managers are arranged and operated one by one on the computers constituting the dispersion processing system, and have functions of constructing a communication path for moving the service agent among the routine managers operating on the computers and managing the communication path for

moving the service agent among names of service entities operating on the computers and the service entities. Moreover, they have functions of judging whether service entities required by the service agent and received from other routine managers through the communication path are available on their own computers, sending the service agent to the service entities requiring them if they are available or moving the service agent to routine managers on other computers if they are unavailable. The movement among the service entities needed for the cooperative processing of the service entities is carried out by moving the service agent to the service entities assigned by the utilization orders according to the utilization orders of the service agent by using these functions.

[0029]

[Functions] The present invention can give the cooperative processing among application programs in a format independent of application program bodies being operation instruction (service agent) by taking the cooperative processing method among application programs as the above constitution. The cooperative interfaces for cooperative processing are added to the application program bodies, and the cooperative interface can easily change cooperative processing procedures among application pro-

grams through the service agent by executing utilization procedures held by the service agent.

[0030] The routine managers as a movement control means operating on each computer constituting the dispersion processing system established a communication path for sending/receiving the service agent and manage a communication path for sending/receiving the service agent between the names of service entities operating on the computers operated by the routine managers and the service entities. If the service entities are started, a communication path for sending/receiving the service agent among the routine managers on the computers operated by the service entities (application programs + cooperative interfaces) is established, names of the service entities are sent to the routine managers, the routine managers receiving the names manage a communication path to the names of delivered service entities and their service entities. The routine managers judge whether the service entities assigned by utilization orders of service agent received from the routine managers operating on other computers through the communication path are available on their own operating computer, sending the service agent to the pertinent service entities if they are available or moving the service entities to routine managers on other computers if they are unavailable.

[0031] If the service entities receive the service agent through the communication path, the cooperative interfaces acquire procedures and data used by the service entities from the utilization procedures held by the service agent, control the processing of application program bodies having the service entities by executing the acquired procedures, obtain processing results needed by the service agent, send the results to the service agent and store them in data items of the service agent. When it is assigned that programs corresponding to the procedures acquired from the service agent are on another computer, a cooperative interface copies the program on computers where the service entity operates and executes the procedures required by the service agent. After a processing result needed by the service agent is stored, the cooperative interface of the service entity renews the movement destination of the service agent and sends the service agent to a routine manager through the communication path. The routine manager judges whether the service entity assigned by the utilization order of the service agent sent from the service entity is available on its own operating computer, sends the service agent to the service entity if the entity is available or moves the service agent to a routine manager on another computer if the entity is unavailable.

[0032] Thus, the routine manager performs the cooperative processing method among application programs by moving a service entity assigned by utilization procedures constituting a dispersion processing system to a computer where the service entity operates according to utilization orders of a service agent, and the service entity acquires and executes a procedure held by a received service agent.

[0033] In the present invention, the cooperative processing among application programs can be easily changed through the service agent by giving a cooperative processing procedure among application programs in a format independent of an application program body (service agent), adding a cooperative interface for the cooperative processing to the application program body and

/7

executing the utilization procedures of the service agent held by this cooperative interface.

[0034] By using this cooperative processing method, the utilization of combining plural application programs made by operators, operations of the plural application programs made at a request of each operator and acquisition of the result are automatized before by describing utilization procedures combining various application programs necessary for advancing business and their operations as utilization procedures of applica-

tion programs used in a series operation procedures in the service agent, moving this service agent among various service entities operating on computers connected to a computer network and executing utilization procedures described in the service agent by the service entities received by the service agent, and the operating load of each operator can be reduced and the flow of operation can be smoothened by systematizing the utilization procedures of application programs necessary for a series of operations.

[0035] A program for preparing a production schedule, a program for simulating operations of a production system, etc. are considered as application programs suitable in the present invention. When production processes in a plant composed of plural production equipment controlled by computers are managed, respectively, there were such problems that "How many products (a lot) are in the plant?" and then "What equipment and what processing operations are performed?" must be accurately instructed for some products, technicians must make such instructions before by judging production progress conditions changing from time to time, thus it is complicated and has bad efficiency. The present invention enables to autonomously (independent of hands) make the judgments and instructions as described above while exchanging information among equipment providing services

and lots receiving it, thus this is suitable for the autonomy of the production system as described above. The present invention is also applicable to various services (on-line shopping, reservation of tickets, etc.) through a computer network.

[0036]

[Actual Examples] An actual example of the present invention will be described in detail by drawings below. In the present invention, a cooperative processing method among application programs in which processing is made progress will be described by cooperating application programs through plural service entities while moving a service agent among plural service entities composed of application program bodies and cooperative interfaces.

[0037] Fig. 1 is a diagram showing the constitution of a dispersion processing system operated by the cooperative processing method among application programs of this actual example. In this actual example, when some operations are performed by users in the above dispersion processing system, they must be processed by cooperating application programs AP1 → AP2 → AP3 or AP4 in this order. In the processing of each application program, AP1 is processed with data stored in item 1, item 2 of data items of a service agent (operation instruction: described later by Fig. 3) and the processing result is stored in item 6;

AP2 is processed with data stored in item 3, item 4 and the processing result is stored in item 7; AP3 is processed with data stored in item 4, item 6 and the processing result is stored in item 8; and AP4 is processed with data stored in item 5 and the processing result is stored in item 9.

[0038] This dispersion processing system comprises computers 1, a network 0 for performing data among the computers, application program bodies 3 operating on the computers, cooperative interfaces 30 for performing processing in cooperation with other application programs, service entities 4 being processing units as basis of cooperative processing among application programs for giving cooperative interfaces 30 to the application program bodies 3, a service agent 20 moving among the application programs to realize the cooperative processing, and routine managers 10 establishing and managing a communication path for moving the service agent 20 and a communication path among service entities and moving the service agent 20 among the service entities 4.

[0039] In the description of this actual example, the names of computers 1 are C1, C2, C3, respectively, the names of application program bodies 3 are AP1, AP2, AP3, AP4, respectively, and the names of service entities 4 having the application program bodies AP1 ~ AP4 are SE1, SE2, SE3, SE4, respectively.

When computers C1 ~ C3 must be differentiated, symbols 1a, 1b, 1c are used; when application programs AP1 ~ AP4 must be differentiated, symbols 3a, 3b, 3c, 3d are used; when service entities SE1 ~ SE4 must be differentiated, symbols 4a, 4b, 4c, 4d are used.

[0040] A constitutional example of computer 1 is shown in Fig. 2. The computer 1 comprises an input unit 13 composed of a keyboard 11 and a mouse 12, etc., a processing unit 16 being a computer main body stored with a CPU 14 and a memory 15, a display unit 17 displaying data, an external storage unit 18 and a printer 19 printing data.

[0041] The input device 13 may also be a touch panel other

/8

than the above. A scanner for inputting image data into the input unit 13 and a mike inputting voice may also be added. The mouse of input unit 13 is a means for assigning a selected target from a menu including some selection legs displayed on the display unit 17 or may be another unit having such a assigning means, e.g., an optical pen or a touch panel. A speaker outputting voice may also be added as a display unit 17.

[0042] The network 0 is a means sending and receiving data among the plural computers 1, and it may also be any network of LAN (Local Area Network) being a network in a specific place,

WAN (Wide Area Network) being a network among bases, an internet being a network connecting networks with each other.

[0043] One routine manager 10 shown in Fig. 1 exists in each computer and is started at the start-up of computer. For example, a method using a UNIX modem, etc. is given as a method for starting the routine manager 10 at the start-up of computer.

[0044] In the cooperative processing method among application programs based on this actual example, cooperative processing among application programs is carried out by moving the service agent 20 among the computers 1 connected to the network 0 through the routine managers 10, executing and controlling the processing of application program bodies 3 therein through the cooperative interfaces 30 of service entities 4 operating on these computers 1. Thus, a series operations are automatized and supported by performing the control and management of execution of the application program bodies 3 through the plural service entities 4 using the service agent 20 and cooperatively processing among the application programs used in a series of operations.

[0045] The cooperative processing method among application programs based on this actual example can be used by service entities 4 having two or more application program bodies 3 operating on one or more computers 1, but the cooperation of

three computers and four service entities 4 operating on the respective computers is described as an example in Fig. 1. Thus, plural service entities 4 can be included on one computer 1. The number of computers 1 and the number of service entities 4 having application program bodies 3 are not restricted to the illustrated constitution.

[0046] The structure of a typical service agent 20 used in the cooperative processing method among application programs based on this actual example is shown in Fig. 3. Different plural service agents 20 can flow in a system simultaneously. The service agent 20 is prepared by individual users on any computer. In addition, the prepared service agent 20 sometimes also generates a new service agent as necessary.

[0047] As shown in Fig. 3, the service agent 20 comprises an identifier 201 for differentiating individual service agents 20, a movement destination list summary 202 for describing a summary of movement destinations of service agent, a movement list summary 203 for managing paths through which the service agent moves, a movement destination name 204 for showing names of service entities 4 held by application program bodies 3 needed by the service agent 20 at the current point of time and names of computers where the service entities 4 operate, and an all data item 21 for listing all data items 211 ~ 219 needed by

the service entities 4 performing the cooperative processing, a data input procedure 22 for describing an input method corresponding to the data items 211 ~ 219 in the all data item 21, a data output procedure 23 for describing output methods corresponding to the data items 211 ~ 219 in the all data item 21 and service entities 4, an input data processing procedure 24 which is methods for processing data inputted by the data input procedure 22, a movement procedure 25 which is methods for determining a movement destination of this service agent 20 and sending the service agent 20 to the determined moving destination, a control procedure 26 which is a procedure for controlling the execution of application program bodies 3 having the service entities 4, and a generation procedure 27 which is a procedure for generating a new service agent.

[0048] In the structure of service agent 20, the identifier 201, movement destination list summary 202, movement list 203 and movement destination name 204 are items prepared certainly in the preparation of service agent 20. In addition, the items and procedures of all data item 21, data input procedure 22, data output procedure 23, input data processing procedure 24, movement procedure 25, control procedure 26 and generation procedure 27 are different elements in each cooperative processing

method among service entities 4 carried out by the service agent 20, and all the elements do not always exist.

[0049] The all data item 21 comprises inputted data item

/9

270 being data items (211 ~ 215) holding some values, an uninputted data item 280 being data items (216 ~ 219) holding no values. When the inputted data item 270 and the uninputted data item 280 are received in some service entity 4 by the service agent 20, an output data item 271 which is a data item storing data to be transferred to the service entity 4 and an input data item 281 which is a data item storing execution result data of the service entity 4. The input and output mentioned here are viewed from the service agent 20, the delivery of data from the service agent 20 to the service entity 4 is taken as an output and its reverse is taken as an input.

[0050] For example, if data needed in execution of the application program body AP1 is stored in the item 1 (211), item 2 (212) and the execution result is stored in item 6 (216), the item 1 (211) and item 2 (212) are output data items (271) and the item 6 (216) is an input data item (281) in the service entity SE1. If data needed in execution of the application program body AP2 is stored in the item 3 (213), item 4 (214) and the execution result is stored in item 7 (217), the item 3 (213)

and item 4 (214) are output data items (271) and the item 7 (217) is an input data item (281) in the service entity SE2. Similarly, if data needed in execution of the application program body AP3 is stored in the item 4 (214), item 6 (216) and the execution result is stored in item 8 (218), the item 4 (214) and item 6 (216) are output data items (271) and the item 8 (218) is an input data item (281) in the service entity SE3; if data needed in execution of the application program body AP4 is stored in the item 5 (215) and the execution result is stored in item 9 (219), the item 5 (215) is an output data item (271) and the item 9 (219) is an input data item (281) in the service entity SE4. Thus, the output data items (271) and the input data items (281) are different for each service entity received by the service agent 20.

[0051] The data input procedure 22 has an input data selection procedure 221 which is a procedure for selecting input data items 281 corresponding to the application program bodies 3 held by the service entities 4 from the uninputted data items 280, an input data acquisition procedure 222 which is a procedure for acquiring values form a picture by a user, a data acquisition procedure 223 which is a procedure for acquiring output result data of application program bodies 3, an input data conversion procedure 224 which is a procedure for converting the output

result data of application program bodies 3 acquired by this data acquisition procedure 223 to a data format of input data items 281, and an input data storage procedure 225 which is a procedure for storing values acquired by the input data acquisition procedure 222 and values converted in the data format of input data items 281 to the input data item 281.

[0052] 「Conversion of data format」 mentioned here means the conversion of data format. As specific examples, when some data have attribute names 「number」, 「unit price」, application programs output 「10」 and 「30」 as values of 「number」 and 「20」 and 「40」 as values of 「unit price」 in an 「output format of execution results of application programs」 as shown in Table 1 and store the values in the input data item in an 「format of input data items of service agent」 as shown in Table 2, the input data conversion procedure performs a conversion of format from the 「output format of execution results of application programs」 to the 「format of input data items of service agent」 as below.

[0053]

[Table 1]

≅ 「Output format of execution results of application programs」

(((number unit price)

(10 20))

((number unit price)

(30 40)))

[0054]

[Table 2]

≅ [Format of input data item of service agent]

((number unit price)

(10 20))

(30 40))

[0055] The data output procedure 23 has an input data selection procedure 231 which is a procedure for selecting output data items 281 corresponding to the application program bodies 3 held by the service entities 4 from the inputted data items 270, an output data conversion procedure 232 which is a pro-

/10

procedure for converting values output data items 281 selected by this output data selection procedure, a parameter setup procedure 233 which is a procedure for transferring values of output data items 281 converted by the output data conversion procedure 232, and an output data display procedure 234 which is a procedure for displaying the values of output data items 281 converted by the output data conversion procedure 232 on the picture.

[0056] The input data processing procedure 24 is a procedure for processing the data acquired by the input data acquisition procedure 222 and the data acquisition procedure 223. For example, processing for calculating the sum of plural data acquired by the input data acquisition procedure 222, etc. are described as this input data processing procedure 24.

[0057] The movement procedure 25 is a procedure for determining a movement destination of service agent 20 and sending the service agent 20 to the determined movement destination. A method for determining the next movement destination according to a movement order given in the preparation of service agent and a method for determining a movement destination of service agent in accordance with any combinations of data items in the inputted data item 270 and the uninputted data item 280 are given as methods for determining the movement destination of the movement procedure 25. Namely, if the next movement destination is determined according to a movement order given in the preparation of service agent, acquire head elements of the movement destination list summary 202 of service agent 20, store the elements in the movement destination name 204, renew values of the movement destination list summary 202, and send the renewed service agent 20 to the routine managers 10 through an input/output channel 403 described later. If the next movement desti-

nation is determined in accordance with any combinations of data items, as shown in Fig. 21, examine values of noticed data items, store the movement destination determined by the values in the movement destination name 204, and send the service agent 20 with renewed movement destination name to the routine managers 10 through the input/output channel 403.

[0058] The control procedure 26 is a procedure for controlling the execution of application program bodies 3 executed by the cooperative interfaces 30 of service entities 4 and has a control script which is a combination of start and end of application program bodies 3, command execution of application program bodies 3 and plural commands.

[0059] The generation procedure 27 is a procedure for generating a new service agent 20 and has a control script generating the service agent 20.

[0060] Fig. 4 ~ Fig. 6 show a connection form of a logic communication path constructed by the routine managers 10 operating on each computer. The routine managers 10 perform establishment and maintenance of the logic communication path among the plural computers and establishment and maintenance of the input/output channel (Fig. 6: 403) among the service entities 4. The logic communication path is constructed from a combination of channels 40 which are logic communication paths connecting two

mutually connected routine managers 10. Fig. 4 shows that RM1 and RM2, RM2 and RM3 are connected among three routine managers 10 operating on computers C1, C2, C3 to construct an RM1-RM2-RM3 logic communication path. Fig. 5 is a diagram shows a state that a new routine manager 10 is connected to the routine manager 10 RM2 constructing the logic communication path shown in Fig. 4 to change the construction of logic communication path. Fig. 6 shows a connected state that the routine manager 10 RM3 and the service entities 4 SE3, SE4 in some computer are connected by the input/output channels 403 and a connected state that the cooperative interfaces 30 and the application program bodies 3 constructing the service entities 4 are connected by an application interface 38. The input/output channels 403 are communication paths connecting the routine managers 10 and the service entities 4 and are actually connected between the cooperative interfaces 30 in the service entities 4 and the routine manager 10. The application interface 38 is an interface connecting the input and output of the cooperative interfaces 30 generated in the service entities 4 and the application program bodies 3.

[0061] The routine managers 10 have channel ports 401 as sockets for connecting the channels 40. The channels 40 have channel numbers for identifying individual channel ports 401. The channel numbers are given in the generation of channel ports

401 by the routine managers 10. Namely, the channels 40 are communication paths connecting the channel ports 401 of two routine managers 10.

[0062] If the channel ports 401 are used by generating a channel 40, the routine managers 10 generates one new channel port 402. Therefore, the routine managers 10 always hold a vacant channel port 402 not connected with others. When a connection request is made from another routine manager 10, this vacant channel port 402 is used to generate a channel 40 among the routine managers 10.

/11

[0063] When the vacant channel port 402 generated by the routine managers 10 connects a service entity 4 and a routine manager 10, it is similarly used as in the case of connecting routine managers 10 with each other. Namely, if the input/output channels 403 are generated between the routine manager 10 and the service entities 4, a new vacant channel port is generated in the cooperative interfaces 30 of service entities 4 when the service entities 4 are started, a request for connection from the service entities 4 to the routine manager 10 is made, the routine manager 10 prepares the input/output channels 403 between the vacant channel port 402 and the vacant channel ports of cooperative interfaces 30 of service entities 4, and only the

routine manager 10 generates a new vacant channel port 402. If the channels 40 are thus generated, the routine managers 10 manages the channel ports so that one new vacant channel port 402 is generated and one vacant channel port 402 always remains.

[0064] The service entities 4 generate the vacant channel ports 402 for generating the input/output channels 403 between the service entities 4 and the routine managers 10 at the start, in addition to the generation of input/output channel 403 between the service entities 4 and the routine managers 10, the service entities 4 hold the cooperative interfaces 30 and the application interfaces 38 for connecting the cooperative interfaces 30 and the input/output of application program bodies 3, send the input/output between the cooperative interfaces 30 and the application program bodies 3, i.e., send commands from the cooperative interfaces 30 to the application program bodies 3 by this application interfaces 38, and receive execution result data of the application program bodies 3 in the cooperative interfaces 30.

[0065] The channel ports 401 have a buffering function during reception and hold the data sent through the channels 40 and the input/output channels 403 until a reading request exists.

[0066] Fig. 7 shows a structure of programs in the processing unit 16 for realizing the routine manager 10. The routine

manager 10 is composed of a management table preparation program 41 placed on a memory 15 of processing unit 16, a connection request program 42, a connection management program 43, a movement management program 44, a connection waiting program 45, a connection destination management table 46, a service entity management table 47, and a connection management table 48.

[0067] If the routine manager 10 placed on the memory 15 of processing unit 16 is started, first, the management table preparation program 41 reads a name summary of connection destination computers stored in the external storage unit 18 and prepares the connection destination movement management table 46 on the memory 15 of processing unit 16. Next, it reads a name summary of operating service entities by the computers operated by the routine manager 10 and prepares the service entity management table 47 on the memory 15 of processing unit 16.

[0068] The connection request program 42 acquires connection destination computer names in order from the head of a summary of computer names stored in the connection destination movement management table 46 prepared on the memory 15 of processing unit 16 and requests a connection to the connection destination computers. If the connection cannot be made, it acquires the next connection destination computer name and successively makes connection requests until the connection can be made. If

the connection cannot be made even a connection request is made, the connection request program 42 outputs an error message and ends the processing.

[0069] If the connection management program 43 receives such a report that the connection with the computers (connection destination computers) connected by the connection request program 42 is completed, the connection management program 43 sends the names of computers making the connection requests (connection source computers) to connection destination computers and waits the names of connection destination computers transmitted from the connection destination computers. If the names of connection destination computers sent from the connection destination computers are received, store pairs of the names of connection destination computers and channel port numbers of the channels 40 connected to the connection destination computers in the connection management table 48. If the control is transferred by receiving the channel port numbers, the connection waiting program 45 described later waits for the names of connection destination computers or names of service entities 4 sent from the channel port numbers. If the names of connection destination computers or names of service entities 4 are received, store pairs of the numbers and the channel port numbers of a logic communication path therebetween in the connection management

table 48. If the delivered names are names of connection source computers, transmit the connection destination computer names to the connection source computers.

[0070] If the connection management program 43 stores the pairs of the numbers of connection source computers and the channel port numbers in the connection management table 48 or connection requests are not made at an interval of given time, the movement management program 44 is started. The movement management program 44 successively acquires channel port numbers from the head of connection management table 48, reads the ser-

/12

vice agent 20 from the channel 40 connected to the channel port numbers 401 and checks whether the names of service entities 4 assigned by the movement destination name 204 of read-out service agent 20 are stored in the connection management table 48. When the names are stored, send the service agent 20 to the service entities 4. If the names are stored, send the service agent 20 to the service entities 4. If the names are not stored, retrieve other connected routine managers 10 from the connection management table 48 and send the service agent 20 to resultant routine manager 10. If no service agent 20 is read from the channel 40, read the service agent 20 similarly from the channel 40 connected to the next channel port number 401. Repeat it

until the channel port numbers 401 stored in the connection management table 48 disappear. If all the channel port numbers stored in the connection management table 48 are successively read out, transfer the control to the next connection waiting program 45.

[0071] The connection waiting program 45 waits for a connection request from other routine managers 10 or service entities 4 (wrong number "30", translator) for some given time. If a connection request exists at an interval of some given time, connect the channel 40 among the routine managers 10 of the connection request source or connect the input/output channels 403 among the service entities 4 of the connection request source, generate a new vacant channel port 402, and send the channel port number of channel port 401 of newly connected channel 40 to the connection management program 43. If no connection request is made in an interval of some given time, transfer the control to the movement management program 44.

[0072] Fig. 8(a) shows the structure of connection destination management table 46, Fig. 8(b) shows the structure of service agent management table 47, and Fig. 8(c) shows the structure of connection management table 48.

[0073] The connection destination management table 46 has a connection destination computer name field 460 keeping a name

list of connection destination computers. The connection destination computer name field 460 has computer names 461 which are names of computers making a connection request at the start of routine managers 10 as values. Information of own computer is certainly not included in this connection destination management table 46.

[0074] The service entity management table 47 has a service entity name field 470 keeping a name summary of service entities 4 operating on computers operated by the routine managers 10. The service entity name field 470 has names 471 of service entities 4 operating on computers operated by the routine managers 10 as values.

[0075] The connection management table 48 has a connection destination name field 480 keeping the names of computers operated by other routine managers 10 connected by the channel 40 and the names of service entities 4 connected by the input/output channels 403, a channel port number field 481 keeping channel port numbers of channel ports 401 connected by the channel 40 and the input/output channels 403 used for connection with these connection destinations, and a connection destination kind field 484 for differentiating whether the connection destinations are routine managers 10 or service entities 4. The connection destination name field 480 has connection destination

names 482 connected to the routine managers 10 by the channel 40 and the input/output channels 403 as values, and the channel port number field 481 hold channel port numbers 483 connected by the channel 40 and the input/output channels 403 corresponding to the connection destination names 482. A connection destination kind field 484 takes either values of routine managers 10 showing that the connection destinations of channels are routine managers 10 and service entities 4 showing that the connection destinations of channels are service entities. These two values are put together and called as connection destination kind names 485.

[0076] Fig. 9 shows a structure of programs in the processing unit for realizing the cooperative interface 30. The cooperative interface 30 is composed of programs of a data preparation program 31, an output data program 32, an execution control program 33, a data acquisition program 34, a data storage program 35, a movement destination operating program 36, a function control program 37 and a function table 371 placed on the memory 15 of processing unit 16. The programs need not to be kept in the service agent 20 itself by providing the function table 371, thus the load during the transmission of service agent 20 among the computers can be reduced.

[0077] If the service agent 20 is received through the input/output channels 403, the cooperative interface 30 transfers the received service agent 20 to the data preparation program 31.

[0078] The data preparation program 31 selects the output data item 271 which is a data item storing data transferred from the inputted data item 270 to the application program body 3 by the output data selection procedure 231 in the received service agent 20.

[0079] If the output data item 271 is determined, prepare data transferred to the application program body 3 or data out-

/13

putted to the picture from data stored in the output data item 271 by the output data conversion procedure 232, attach a tag of either data and transfer the prepared data to the data output program 32.

[0080] If the data is transferred to the data output program 32, judge whether the data output program 32 judges the data are outputted from the tag of received data to the picture or sent to the application program body 3, and it outputs the data to a picture by the output data display procedure 234 of the service agent 20 in case of outputting the data to the screen or transfers the data to the execution control program 3

in case of transferring the data to the application program bodies 3. When the output data display procedure 234 described in the service agent 20 is not a program but simply a function name, the data output program 32 transfers the function name to the function control program 37.

[0081] If the function control program 37 receives the function name, it retrieves a function pertinent to the function name from the function table 371 and executes the retrieved program. If the execution control program 33 receives the data from the data output program 32, it executes the control procedure 26 of service agent 20 and controls the execution of application program body 3. When the function control program 37 executes the application program body 3, it sends the data to the application program body 3 by the parameter setup procedure 233 of service agent 20 in case the data must be transferred to the application program body 3. If the execution of application program body 3 is controlled and the execution of control procedure is ended, acquire the execution result data of application program body 3 by the data acquisition procedure 223 of service agent 20 and transfer the acquired data to the data acquisition program 34. If the control procedure 26 is a function name or processing of only a function name is included in the control

procedure 26, transfer the function name to the function control program 37 and execute a pertinent function program.

[0082] If the data acquisition program 34 receives the execution result data, select the input data item 281 which is a data item storing the execution result data of application program body 3 from the all data item 21 of service agent 20 by the input data selection procedure 221 of service agent 20. When a picture is outputted, acquire only necessary data in the data inputted to the picture by the input data acquisition procedure 222 of service agent 20. Transfer these acquired data (execution result data of application program body 3, input data from the picture) and the input data item 281 to the data storage program 35. If the input data acquisition procedure 222 described in the service agent 20 is not a program but only a simple function name, transfer the function name to the function control program 37 by the data acquisition program 34 and executes the program of function of the pertinent function name.

[0083] If the data storage program 35 receives the data from the data acquisition program 34 and the input data item 281, in case the input data processing procedure 24 exists, process data stored in the received data and the all data item 21 by the input data processing procedure 24 with reference to the region of input data processing procedure 24 of the service

agent 20, convert the data after processing to a data format of received input data item 281 by the input data conversion procedure 224 and store the converted data to respective data items of the input data item 281. In case the input data processing procedure 24 does not exist, convert the data after processing to a data format of received input data item 281 by the input data conversion procedure 224 and store the converted data to respective data items of the input data item 281.

[0084] The movement destination operating program 36 stores the data in the input data item 281 calls it therefrom, acquires head elements of the movement destination list summary 202, stores them in the movement destination name 204, and sends the service agent 20 with a movement destination name 204 renewed in this manner to the routine managers 10 through the input/output channels 403.

[0085] Fig. 10(a) shows a structure of function table 371 used in the function control program 37. The function table 371 comprises a function name field 372 and a processing field 373 which is a processing program body of function. Either a program 373a describing processing contents of function or a storage position of program 373b describing processing contents of the function are described in the processing field 373.

[0086] If the processing field 373 of pertinent function name is the program 373a, the function control program 37 executes the program. If it is the storage position of program 373b, read (copy) the program 374 at the storage position on the memory 15 of processing unit 16 through the network 0 and then execute the program. Programs on other computers can be shared by keeping the storage position of program rather than the program itself in such a function table. It has advantages of reducing the memory capacity of the function table and changing

/14

the program in only one place in case of program change.

[0087] The storage position of program 373b is assigned by the description format of Fig. 10(b). The storage positions are assigned in the form of a computer name 375a stored by the program and a storage pass 375c indicating the storage position of program on a computer assigned by the computer name 375a, and assigned in the form of computer name 375a, breakoff symbol 375b and storage pass 375c by using the breakoff symbol 375b for differentiating the computer name 375a and storage pass 375c.

[0088] Fig. 11(a) shows the description format of the movement destination list 91 stored in the movement destination list summary 202 of the service agent 20, and Fig. 11(b) shows its specific description example.

[0089] The movement destination list 91 connects a movement destination computer name 912 which is names of movement destination computers of the service agent 20 and a movement destination 911 which is a group of service entity names 913 being names of service entities 4 utilized by the movement destination computers with arrows (→) according to a movement order of the service agent 20. When the movement destination computer name are unknown, Unknown is assigned to the movement destination computer name 912.

[0090] In the example of Fig. 11(b), it is shown that the service agent 20 performs processing by an application program body 3 AP1 having a service entity 4 named as SE1 of a computer named as C1, then performs processing by an application program body 3 AP2 having a service entity 4 named as SE2 of a computer named as C2, next performs processing by an application program body 3 AP3 having a service entity 4 named as SE3 of a computer named as C3 and finally performs processing by an application program body 3 AP4 having a service entity 4 named as SE4 although the name of a computer operating the service entity 4 SE4 is unclear.

[0091] Fig. 12(a) shows the description format 101 of the movement destination list stored in the movement list summary

203 of the service agent 20, and Fig. 12(b) shows its specific description example.

[0092] The movement list summary 203 seeks "Which computer on a network does a service utilized by the service agent 20 operates on?" and is used in movement. More specifically, the movement list summary 203 is used when the movement destination computer name 912 of the movement destination list shown in Fig. 11 is Unknown. If the movement destination computer name 912 is Unknown, the routine manager 10 of each computer checks "Whether a service entity 4 having a name assigned by the movement destination name 204 of service agent 20 exists?", if it does not exist, stores the computer name in the movement destination computer name 102 and sends the service agent 20 to the routine manager 10 on another computer.

[0093] When the movement destination computer name 912 is Unknown, the movement list 101 connects the name of computer, where service entity 4 assigned by the movement destination names 204 of service agent 20 does not exist, with arrows (→) according to a movement order of the service agent 20. The movement list 101 successively supplements a computer name every time the service agent 20 moves and cleared at the time of finding a computer where the service entity 4 assigned by the movement destination name 204 of service agent 20 operates.

Thus, computers once sought are stored in this list 101, and many times of retrievals of the same computer can be prevented. The movement list summary 203 is useful to avoid seeking services utilized by the service agent 20 and visiting the same computer many times to perform an effective search of services because there is a possibility of having such an event if this list does not exist.

[0094] The example of Fig. 12(b) shows that the service entity 4 assigned by the movement destination name 204 of service agent 20 does not exist in the computer named as C1 and then also does not exist in the computer named as C2.

[0095] Fig. 13 ~ Fig. 16 show flows of movement processing procedures of service agent 20 performed by the routine managers 10 to realize the cooperative processing method among application programs using the service agent 20.

[0096] The service agent 20 is sent to the routine managers 10 of computer prepared thereby. At this time, the value of movement destination list summary 202 of service agent 20 is a movement order shown in Fig. 11(b). The value of movement destination name 204 of the service agent 20 at this time is taken as the head element SE1 (911a) of movement destination list 91 which is the value of movement destination list summary 202.

[0097] If the service agent 20 is received (1100), first, the routine manager 10 checks whether the movement destination computer name of movement destination name 204 of the service agent 20 is same as the name of computer where the routine manager 10 operates (1101).

[0098] If the movement destination computer name of movement destination name 204 of service agent 20 and the name of computer where the routine manager 10 operates are different, check whether the connection destination in which the connection destination name 482 of connection destination name field 480 is same as the movement destination computer name is registered from connection destinations in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is a routine manager (1102). If the movement destination computer name is registered, acquire a channel port number 483 stored in the channel port number field 481 of the connection destination name 482 pertinent to the movement destination computer name (1103), and send the service agent 20 to the routine manager 10 of movement destination computer through the channel 40 connected to the channel port number 483 (1104). If the movement destination computer name is not registered, acquire a channel port number 483 stored in the channel port number field 481 of the first registered connection desti-

nation name 482 from connection destinations in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is a routine manager (1105), and send the service agent 20 to the routine manager 10 of registration destination computer through the channel 40 connected to the channel port number 483 (1106).

[0099] When the movement destination computer name 912 of movement destination name 204 of the service agent 20 and the name of computer in which the routine manager 10 operates are the same, check whether the service entity in which the connection destination name 482 of connection destination name field 480 is same as the service entity name of movement destination name 204 is registered from connection destinations in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is the service entity (1107). If the service entity name is registered, acquire a channel port number 483 stored in the channel port number field 481 of the registered service entity (1108), and send the service agent 20 to a cooperative interface 30 of service entity 4 through the input/output channels 403 connected to the channel port number 483 (1109).

[0100] If the service entity name is not registered in the connection management table 48, check whether the service entity

name is registered in the service entity management table 47 (1110). If the service entity name is registered in the service entity management table 47, start the service entity 4 (1111). The cooperative interface 30 of the started service entity 4 makes a connection request to the routine manager 10. The routine manager 10 waits for the connection request from the cooperative interface 30 of service entity 4 (1112), if the connection request is received, generate input/output channels 403 between the routine manager 10 and the cooperative interface 30 (1113), register the name of started service entity and the channel port numbers of the input/output channels 403 in the connection management table 48 (1114), and sent the service agent 20 to the cooperative interface 30 of service entity 4 through the generated input/output channels 403 (1109). If the service entity name is not registered in the service entity management table 47, change the movement destination computer name 912 of movement destination name 204 of the service agent 20 to Unknown (1115a), acquire the channel port number 483 stored in the channel port number field 481 of the first registered connection destination name 482 (1105), and send the service agent 20 to the routine manager 10 of registration destination computer connected to a channel through the channel 40 connected to the channel port number 483 (1106).

[0101] If the movement destination computer name 912 of movement destination name 204 of the service agent 20 is Unknown, shift the flow to Fig. 14, check whether the connection destination in which the connection destination name 482 of connection destination name field 480 is same as the service entity name of movement destination name 204 of the service agent 20 is registered from connection destinations in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is the service entity (1115b). If the service entity name is registered, acquire the channel port number 483 stored in the channel port number field 481 of the registered service entity (1116), and send the service agent 20 to the cooperative interface 30 of service entity 4 through the input/output channels 403 connected to the acquired channel port number 483 (1117). If the service entity name is not registered in the connection management table 48, check whether the service entity name is registered in the service entity management table 47 (1118). If the service entity name is registered in the service entity management table 47, start the service entity 4 (1119). The cooperative interface 30

/16

of the started service entity 4 makes a connection request to the routine manager 10. The routine manager 10 waits for the

connection request from the cooperative interface 30 of service entity 4 (1120), if the connection request is received, generates input/output channels 403 between the routine manager 10 and the cooperative interface 30 (1121), register the name of started service entity and the channel port number of the input/output channels 403 in the connection management table 48 (1122), and sent the service agent 20 to the cooperative interface 30 of service entity 4 through the generated input/output channels 403 (1117).

[0102] If the service entity name is not registered in either the connection management table 48 or the service entity management table 47, shift the flow to Fig. 15, check whether a connection destination in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is a routine manager exists (1123). If the connection destination is registered, fetch a first registered connection destination name 482 therein (1124), and check whether the connection destination name 482 is in the movement computer name 102 of movement list 101 stored in the movement list summary 203 (1125). If the connection destination name 482 is not in the movement list 101, supplement the name of current computer to the movement computer name 102 of movement list 101 (1126), store the supplemented movement list 101 in the movement

list summary 203 of service agent 20 (1127), fetch the channel port number 483 stored in the channel port number field 481 of the connection destination name 482 (1128), and send the service agent 20 to the routine manager 10 of a computer connected to the channel through the channel 40 connected to the channel port number (1129). If the connection destination name 482 is in the movement list 101, check whether the next connection destination in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is a routine manager (1123). If the next connection destination is registered in the connection management table 48, fetch the connection destination name 482 (1124) and check whether the connection destination name 482 is in the movement computer names 102 of the movement list 101 stored in the movement list summary 203 of service agent 20 (1125). This is successively repeated until the connection destination in which the movement destination kind name 485 of movement destination kind field 484 of the connection management table 48 is a routine manager 10 disappears or a connection destination name 482 not in the movement list 101 is found. As a result, if the connection destination in which the movement destination kind name 485 of movement destination kind field 484 of the connection management

table 48 is a routine manager 10 disappears, end the movement of service agent 20 as an error (1130).

[0103] Shift the flow to Fig. 16, the cooperative interface 30 of service entity 4 receiving the service agent 20 executes a data output procedure described in the received service agent 20 and acquires data sent from the service agent 20 to the service entity 4 (1131), executes a control procedure 26 and controls the execution of application program body 3 having the service entity 4 (1132), executes a data input procedure 22 of service agent 20 and stores the execution result data in the service agent 20 (1133), executes a movement procedure 25 of service agent 20 and write a value of movement destination name 204 in the next movement destination (1134), and sends the service agent 20 written with the value of movement destination name 204 from the cooperative interface 30 to the routine manager 10 through the input/output channels 403 (1135). Return the flow to Fig. 13, if the routine manager 10 receives the service agent 20 sent from the cooperative interface 30 through the input/output channels 403 (1100), similarly as the case of receiving the service agent 20 through the channel 40, check a movement destination computer name 912 of movement destination name 204 (1101) and send the service agent 20 to the next movement destination through the channel 40 (1104). The service agent 20 is moved

among the service entities 4 by repeating the movement processing of this service agent 20 among routine managers 10 until it becomes the last element of movement destination list 91, and the execution of application program bodies 3 is controlled by processing the moved service agent 20 with the cooperative interfaces 30 of the service entities 4.

[0104] Fig. 17 ~ Fig. 20 show processing flows for controlling the execution of application program bodies 3 while utilizing data and procedures held by the received service agent 20 in the cooperative interfaces 30 of the service entities 4.

[0105] First, in Fig. 17, if a cooperative interface 30 receives the service agent 20 from a routine manager 10 through the input/output channels 403 (1200), execute an output data selection procedure 231 of the received service agent 20 and determine an output data item 271 which is a data item storing data transferred from the inputted data item 270 to application program bodies 3 (1201).

/17

[0106] If the output data item 271 is determined, check whether the output data conversion procedure 232 of the service agent 20 is a program or a function name (1202). If the output data conversion procedure 232 is a program, check whether the output data conversion procedure 232 is a data output function

or a picture output function to the application program bodies 3 (1203); if it is a data output function, execute the function and generate data transferred from data stored in the output data item 271 to the application program bodies 3 (1204). If it is a picture output function, execute the function and prepare data outputted from data stored in the output data item 271 to a picture (1205). Attach a tag indicating which data it is for the generated data (1206). If the output data conversion procedure 232 is a function name, acquire a program of the function name from a function table through the function control program 37 (1207) and check whether the obtained function is an data output function or a picture output function to the application program bodies 3.

[0107] Judge whether the tag-attached data is outputted from the tag to a picture or transferred to the application program bodies 3 (1208), in the case of data outputted to a picture, check whether the output data procedure 234 of service agent 20 is a program or a function name (1209). If the output data procedure 234 is a program, execute the output data procedure 234 and output a picture output data (1210). If the output data procedure 234 is a function name, acquire a program of the function name from the function table through the function control program 37 (1211), execute the acquired function and output

the picture output data to a picture (1210). In the case of data transferred to the application program bodies 3, delete the tag of data (1212).

[0108] Next, shift the flow to Fig. 18 and check whether the control procedure 26 of service agent 20 is a program or a function name (1213). If the control procedure 26 is a program, control the execution of application program body 3 by executing the control procedure 26 (1214). If the control procedure 26 is a function name, acquire a program of the function name from the function table through the function control program 37 (1215) and control the execution of application program bodies 3 by executing the acquired function (1214).

[0109] If the execution control of the application program bodies 3 is ended by the control procedure 26, check whether the data acquisition procedure 223 of service agent 20 is a program or a function name (1216). If the data acquisition procedure 223 is a program, execute the data acquisition procedure 223 and acquire execution result data of the application program bodies 3 (1217). If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1218), execute the acquired function and acquire execution result data of the application program bodies 3 (1217).

[0110] If a picture is outputted, shift the flow to Fig. 20, and check whether the control procedure 26 of service agent 20 is a program or a function name (1219). If the control procedure 26 is a program, control the input/output from the picture by executing the control procedure 26 (1220). If the control procedure 26 is a function name, acquire a program of the function name from the function table through the function control program 37 (1221) and control the input/output from the picture by executing the acquired function (1220).

[0111] If a picture is outputted, check whether the data acquisition procedure 223 of service agent 20 is a program or a function name (1222). If the data acquisition procedure 223 is a program, execute the data acquisition procedure 223 and acquire input data from the picture (1223). Namely, incorporate instructions or data inputted by a user through a picture. If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1224), execute the acquired function and acquire input data from the picture (1223).

[0112] Next, return the flow to Fig. 18, and check whether the input data selection procedure 221 of service agent 20 is a program or a function name (1225). If the input data selection procedure 221 is a program, execute the input data selection

procedure 221 and select an input data item 281 which is a data item storing processing result data of application program bodies 3 from the all data item 21 of service agent 20 (1226). If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1227), execute the acquired function and select an input data item 281 which is a data item storing processing result data of application program body 3 from the total data item 21 of service agent 20 (1226).

[0113] Next, shift the flow to Fig. 19, and check whether the input data processing procedure 24 of service agent 20 is a program or a function name (1228). If the input data processing

/18

procedure 24 is a program, execute the input data processing procedure 24 and processing the received data and data stored in the all data item 21 (1229). If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1230), execute the acquired function and process the received data and data stored in the all data item 21 (1229).

[0114] Next, check the input data conversion procedure 224 is a program or a function name (1231). If the input data conversion procedure 224 is a program, execute the input data con-

version procedure 224, convert the received data to the data format of an input data item 281 pertinent thereto (1232) and store the converted data in respective data items of the input data item 281 (1234). If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1233), execute the acquired function and convert the received data to the data format of an input data item 281 pertinent thereto (1232) and store the converted data in the respective data items of the input data item 281 (1234).

[0115] If the data is stored in the input data item 281, check whether the movement procedure 25 of service agent 20 is a program or a function name (1235). If the movement procedure 25 is a program, execute the movement procedure 25, rewrite the value of movement destination name 204 in the next movement destination (1236) and send the service agent 20 with rewritten the value of movement destination name 204 from the cooperative interfaces 30 to the routine manager 10 through the input/output channels 403 (1237). If the data acquisition procedure 223 is a function name, acquire a program of the function name from the function table through the function control program 37 (1238), rewrite the value of movement destination name 204 in the next movement destination (1236) and send the service agent 20 with

rewritten value of movement destination name 204 from the cooperative interfaces 30 to the routine manager 10 through the input/output channels 403 (1237).

[0116] Next, an example of applying the present invention to a production line management system is described as its specific application example. In the production line management system, production and inspection machines, application programs for control of these machines and application programs necessary for production management must be cooperated in a determined order for each product.

[0117] Fig. 22 shows the system constitution of one application example of a production line management system based on the present invention. In the application example, the production line management system is composed of computers C2 (1342), C3 (1343) for managing production processes A, B, production machines MA1 (1332), MB1 (1333) connected to production processes A, B, application programs AP2 (1302), AP3 (1303) for controlling these production machines MA1 (1332), MB1 (1333), an application program AP1 (1301) for preparing a production instruction on the computer C1 and a network 0 connecting the computers, and the production instruction prepared by the application program AP1 on the computer C1 is described by cooperatively controlling the production machines connected to produc-

tion processes with a case of advancing the production as an example.

[0118] In the production line management system based on this actual example, these various application programs AP1 (1301), AP2 (1302), AP3 (1303) are realized as service entities SE1 (1321), SE2 (1322), SE3 (1323) (name of each service entity is SE1, SE2, SE3, respectively). The production instruction attached to a product is embodied as a service agent SA1 (1400). A user prepares the service entity SA1 (1400) embodying the production instruction by using the service entity SE1 (1321), first, the user transmits a production instruction to the application program AP2 (1302) through the service entity SE2 (1322) having the application program AP2 (1302) for controlling the production machine MA1 (1332) (wrong number "1333" in original document, translator) of production process A, next transmits the production instruction to the application program AP3 (1303) through the service entity SE3 (1323) having the application program AP3 (1303) for controlling the production machine MB1 (1333) of production process B, subsequently returns the flow to the service entity SE1 (1321) with a prepared service agent SA1 (1400).

[0119] Fig. 23 shows the structure of service agent SA1 (1400) which is one example of service agent expressing a pro-

duction instruction at a time that of preparing it on the computer C1 (1341) by using the service entity SE1 (1321). The service agent SA1 (1400) has an identifier ID1 (1401), movement destination list summaries (C2 SE2) (C3 SE3) (C1 SE1) (1402), a movement list summary nil (1403), a movement destination name (C2 SE2) (1404), in addition, a process A production speed item (1405) which is an item being a setup parameter of production machine MA1 and its value 10, a process A required time item (1406) which is an item of data acquired from the production

/19

machine MA1, a process B production speed item (1407) which is an item being a setup parameter of the production machine MB1 and its value 15 and a process B production speed item (1408) which is an item of data acquired from the production machine MB1 as all data item. The service agent SA1 (1400) also has an output data selection procedure (1421), an output data conversion procedure (1422) and a parameter setup procedure (1423) as data output procedures, and a data acquisition procedure (1411), an input data selection procedure (1412) and an input data storage procedure (1413) as data input procedures and further has a control procedure (1430).

[0120] The service agent SA1 (1400) generated by the service agent SA1 (1321) is sent to the computer C1 (1341) through

the channel 40. A routine manager RM1 (1311) checks the value (C2 SE2) (1404) of movement destination name of the service agent SA1 (1400), retrieves the channel 40 connected with the computer C2 (1342) assigned by the movement destination name and sends the service agent SA1 (1400) to the computer C2 (1342) through the resultant channel 40.

[0121] If the routine manager RM2 (1312) on the computer C2 (1342) receives a delivered service agent SA1 (1400), check the value of movement destination name (C2, SE2) (1404), retrieve the channel 40 connected with the service entity SE2 (1322) assigned by the movement destination name and send the service agent SA1 (1400) to the service entity SE2 (1322) through the resultant channel 40. If the service entity SE2 (1322) receives the service agent SA1 (1400) through the channel 40, the cooperative interface 30 of the service entity SE2 (1322) acquires an output data selection procedure 1421 held by the service agent SA1 (1400), execute the procedure 1421 and determine a process A production speed item 1405 which is a data item to be transferred to the application program AP2 (1302) held by the service entity SE2 (1322). Next, acquire an output data conversion procedure 1422 held by the service agent SA1 (1400), execute the procedure 1422, fetch a value 10 of the process A production speed item 1405 which is a selected item, attach a tag indica-

ting that the data is transferred to an application program, then acquire a parameter setup procedure 1423 held by the service agent SA1 (1400), execute the procedure 1423 and set up the value 10 assigned by the application program AP2 (1302). Then, processing of the application program AP2 (1302) is performed by acquiring the control procedure 1430 held by the service agent SA1 (1400) and executing the procedure 1430.

[0122] If the processing of application program AP2 (1302) is ended, acquire the data acquisition procedure 1411 held by the service agent SA1 (1400), execute the procedure 1411 and acquire execution result of the application program. Next, acquire the input data selection procedure 1412 held by the service agent SA1 (1400), execute the procedure 1412, select the process A required time 1406 which is a data item storing the execution result of the application program, then acquire the input data storage procedure 1413 held by the service agent SA1 (1400), execute the procedure 1413, and store the execution result of the application program AP2 (1302) in the process A required time item 1406.

[0123] Next, fetch a second element (C3, SE3) of the movement destination list summary 1402, store it in the movement destination name 1406 and send it to the routine manager RM2 (1312) through the channel 40.

[0124] If the routine manager RM2 (1312) receives the service agent SA1 (1400) sent from the service entity SE2 (1322) through the channel 40, check its movement destination name (C3, SE3) (1404), retrieve the channel 40 connected to a computer C3 (1343) assigned by the movement destination name (C3, SE3) (1404), and send the service agent SA1 (1400) through the resultant channel (40).

[0125] If the routine manager RM3 (1313) on the computer C3 receives the delivered service agent SA1 (1400), check its movement destination name (C3, SE3) (1404), retrieve the channel 40 connected to the service entity SE3 (1323) assigned by the movement destination name (C3, SE3) (1404), and send the service agent SA1 (1400) to the service entity SE3 (1323) through the resultant channel (40).

[0126] If the service entity SE3 (1323) receives the service agent SA1 (1400) through the channel 40, the cooperative interface 30 of the service entity SE3 (1323) acquires the output data selection procedure 1421 held by the service agent SA1 (1400), executes the procedure 1421 and determines a process B production speed item 1407 which is a data item to be transfer-

/20

red to the application program AP3 (1303) held by the service entity SE3 (1323). Next, acquire the output data conversion pro-

cedure 1422 held by the service agent SA1 (1400), execute the procedure 1422, fetch a value 15 of the process B production speed item 1407 which is a selected item, attach a tag indicating that the data is transferred to the application program, then acquire the parameter setup procedure 1423 held by the service agent SA1 (1400), execute the procedure 1423 and set up the value 15 assigned by the application program AP3 (1303). Then, processing of the application program AP3 (1303) is performed by acquiring a control procedure 1430 held by the service agent SA1 (1400) and executing the procedure 1430.

[0127] If the processing of application program AP3 (1303) ends, acquire the data acquisition procedure 1411 held by the service agent SA1 (1400), execute the procedure 1411 and acquire execution result of the application program. Next, acquire the input data selection procedure 1412, execute the procedure 1412, select the process B required time item 1408 which is a data item storing execution result of the application program, then acquire the input data storage procedure 1413 held by the service agent SA1 (1400), execute the procedure 1413, and store execution result of the application program AP3 (1303) in the process B required time item 1408.

[0128] Next, fetch the (C1 SE1) which is the third element of the movement destination list summary 1402, store it in the

movement destination name 1404 and send it to the routine manager RM3 (1313) on the computer C3 (1343) through the channel 40.

[0129] If the routine manager RM3 (1313) receives the service agent SA1 (1400) sent from the service entity SE3 (1323) through the channel 40, check the movement destination name (C1 SE1) (1404) and retrieve the channel 40 connected to the computer C1 (1341) assigned by the movement destination name (C1 SE1) (1404), but the computer name C1 (1341) is not in the computer connected to the routine manager RM3 (1313) on the computer C3 (1343), therefore send the service agent SA1 (1400) to the routine manager RM2 (1312) of connected computer C2 (1342).

[0130] If the routine manager RM2 (1312) on the computer C2 (1342) receives the delivered service agent SA1 (1400), check the value of movement destination name (C1 SE1) (1404), the computer name C1 assigned by the movement destination name 1404 is different from the computer name C2, therefore store the computer name C2 in the movement list 1403 of the service agent SA1 (1400), take the value of movement list 1403 as ((C2)), retrieve the channel 40 connected to the computer name C1 assigned by the movement destination name (C1 SE1) (1404) of the service agent SA1 (1400) and send the service agent SA1 (1400) to the resultant channel 40.

[0131] If the routine manager RM1 (1311) receives the service agent SA1 (1400) sent from the routine manager RM1 (1311) through the channel 40, check the movement destination name (C1 SE1) (1404), check whether the movement destination name 1404 and the computer name C1 are the same, then retrieve the channel 40 connected to the service entity SE1 (1321) assigned by the movement destination name (C1 SE1) (1404), and send the service agent SA1 (1400) to the service entity SE1 (1321) through the resultant channel 40.

[0132] In this manner, the application programs AP2 (1302), AP3 (1303) for controlling the manufacturing machine MA1 (1332) connected to the computers C2 (1342), C3 (1343) managing the manufacturing processes A, B can be cooperatively controlled. Even if the manufacturing processes are different, manufacturing equipment for control and their order can be changed only by changing the summary of movement destinations assigned by the movement destination list summary, thus the cooperative method can be easily changed.

[0133] As described above, the present invention enables cooperative processing among flexible application programs by giving cooperative interfaces for the cooperative processing to application program bodies, describing cooperative processing procedures necessary for the cooperation and data necessary at

the time of cooperation, and storage data items and control procedures for execution of individual application program bodies in an operation instruction and moving the operation instruction among computers in order to flexibly process various application programs.

[0134] Thereby, the present invention enables to change the cooperative processing procedures among application programs by the operation instruction without stopping the execution of application programs as targets of the cooperative processing. Moreover, a user can utilize the application programs by assign-

/21

ing application programs to be utilized without being aware of "By which computer on a dispersion system are application programs to be cooperatively processed operated?".

[0135] The present invention also enables to automatize a series operations depending upon many operators and made by operating application programs by the operators.

[Brief Description of the Drawings]

[Fig. 1] Block diagram showing the constitution of actual example of a dispersion processing system performing cooperative processing among application programs based on the present invention.

[Fig. 2] Block diagram showing the construction of a computer shown in Fig. 1.

[Fig. 3] Illustrative diagram showing the structure of a service agent in the actual example.

[Fig. 4] Illustrative diagram of a connected state of communication path among routine managers in the actual example.

[Fig. 5] Illustrative diagram showing the connected state of a new routine manager with respect to the connected state of Fig. 4.

[Fig. 6] Illustrative diagram showing the connected state of routine managers and service entities in the actual example.

[Fig. 7] Block diagram showing the program structure of a routine manager in the actual example.

[Fig. 8] Illustrative diagram of the structure of tables shown in Fig. 7.

[Fig. 9] Block diagram showing the program structure of a cooperative interface 30 in the actual example.

[Fig. 10] Illustrative diagram of the structure of function table 371 shown in Fig. 9.

[Fig. 11] Illustrative diagrams of description format and description example of a movement destination list 91 of a movement destination list summary 202 shown in Fig. 3.

[Fig. 12] Illustrative diagrams of description method and description example of a movement list 101 of a movement destination list summary 203 shown in fig. 3.

[Fig. 13] Flow chart (1) showing a movement processing procedure of the service agent in the actual example.

[Fig. 14] Flow chart (2) showing a movement processing procedure of the service agent in the actual example.

[Fig. 15] Flow chart (3) showing a movement processing procedure of the service agent in the actual example.

[Fig. 16] Flow chart (4) showing a movement processing procedure of the service agent in the actual example.

[Fig. 17] Flow chart (1) showing processing of a cooperative interface in the actual example.

[Fig. 18] Flow chart (2) showing processing of a cooperative interface in the actual example.

[Fig. 19] Flow chart (3) showing processing of a cooperative interface in the actual example.

[Fig. 20] Flow chart (4) showing processing of a cooperative interface in the actual example.

[Fig. 21] Illustrative diagram of an example of a movement destination determining method based on combinations of data items in the actual example.

[Fig. 22] Block diagram showing a constitutional example of a production line management system applied with the present invention.

[Fig. 23] Illustrative diagram showing a structural example of a service agent in the system of Fig. 22.

[Description of the Symbols]

1] computer

3] application program body

4] service entity

10] routine manager

20] service agent

30] cooperative interface

【図2】

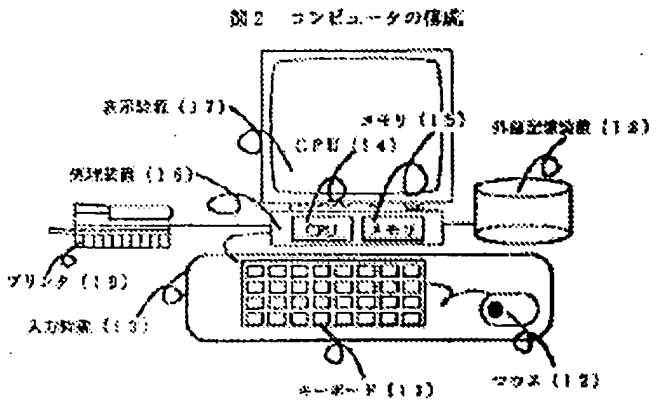


Fig. 2 Construction of computer

11] keyboard

12 ⌋ mouse

13 ⌋ input unit

14 ⌋ CPU

15 ⌋ memory

16 ⌋ processing unit

17 ⌋ display unit

18 ⌋ external storage unit

19 ⌋ printer

【図4】

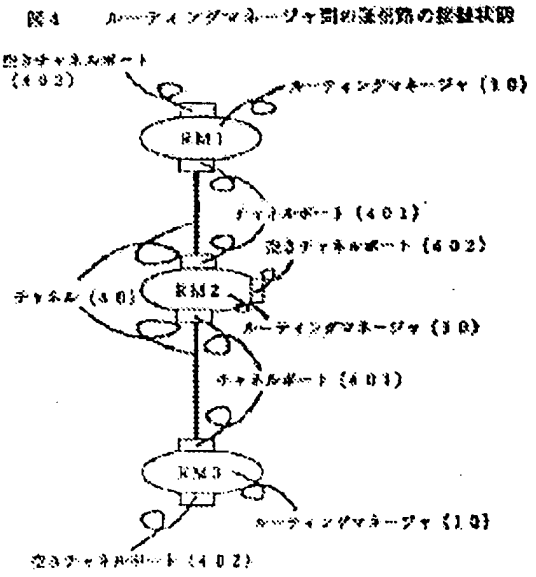


Fig. 4 Connected state of communication path among routine managers

10 ⌋ routine manager

40) channel

401) channel port

402) vacant channel port

/22

【図1】

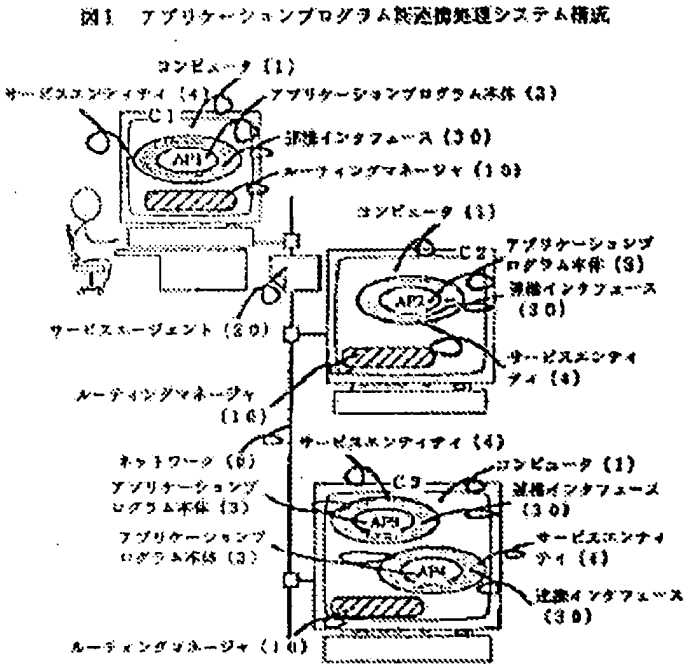


Fig. 1 Constitution of cooperative processing system among application programs

0] network

1] computer

3] application program body

4] service entity

10] routine manager

20] service agent

30] cooperative interface

【図3】

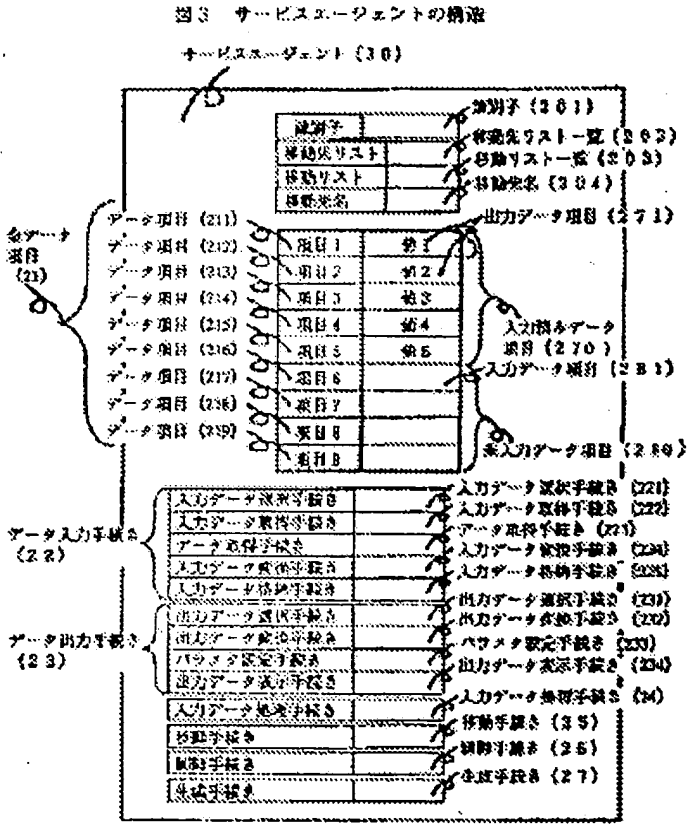


Fig. 3 Structure of service agent

20] service agent

21] all data item

22] data input procedure

23] data output procedure

24] input data processing procedure

25] movement procedure

26] control procedure

27] generation procedure

201] identifier

202] movement destination list summary

203] movement list summary

204] movement destination name

211] data item

212] data item

213] data item

214] data item

215] data item

216] data item

217] data item

218] data item

219 ⌋ data item

221 ⌋ input data selection procedure

222 ⌋ input data acquisition procedure

223 ⌋ data acquisition procedure

224 ⌋ input data conversion procedure

225 ⌋ input data storage procedure

231 ⌋ output data selection procedure

232 ⌋ output data conversion procedure

233 ⌋ parameter setup procedure

234 ⌋ output data display procedure

(table)

Identifier	
Movement destination list	
Movement list	
Movement destination name	

Item 1	Value 1
Item 2	Value 2
Item 3	Value 3
Item 4	Value 4
Item 5	Value 5

Item 6	
Item 7	
Item 8	
Item 9	

Input data selection procedure	
Input data acquisition procedure	
Data acquisition procedure	
Input data conversion procedure	
Input data storage procedure	
Output data selection procedure	
Output data conversion procedure	
Parameter setup procedure	
Output data display procedure	
Input data processing procedure	
Movement procedure	
Control procedure	
Generation procedure	

【図5】

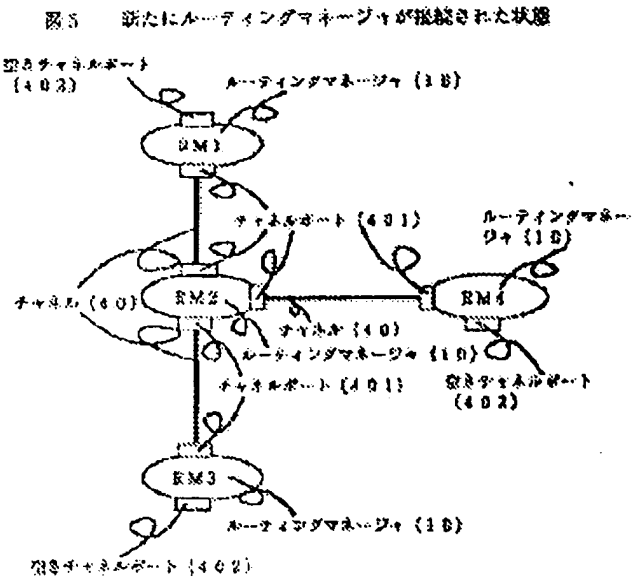


Fig. 5 Connected state of new routine manager

10] routine manager

40] channel

401] channel port

402] vacant channel port

【図6】

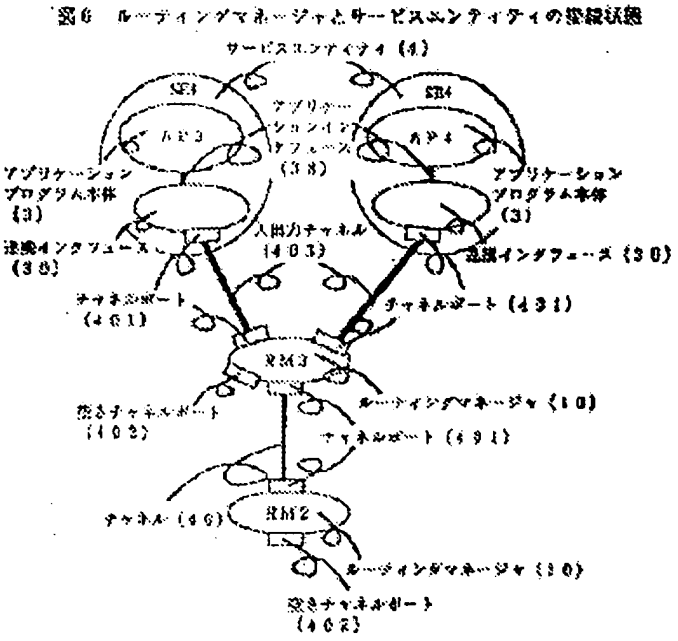


Fig. 6 Connected state of routine managers and service entities

3] application program body

4] service entity

10] routine manager

30] cooperative interface

38] application interface

40] channel

401] channel port

402] vacant channel port

403] input/output channel

【図7】

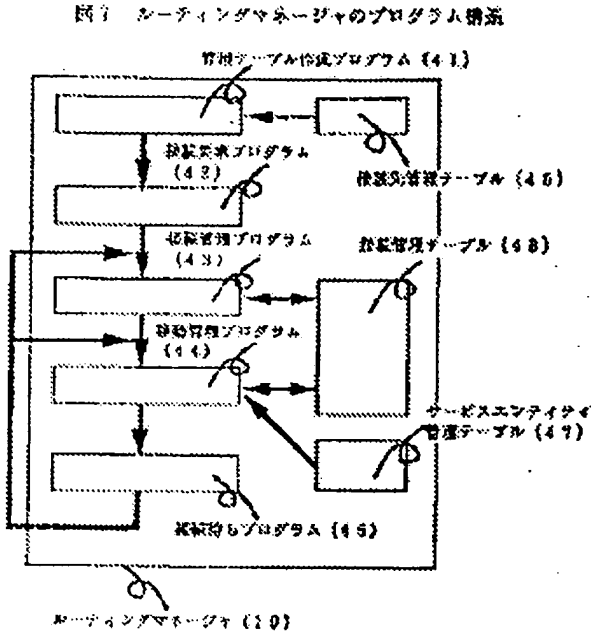


Fig. 7 Program structure of routine manager

10] routine manager

41] management table preparation program

42] connection request program

43] connection management program

44] movement management program

45] connection waiting program

46] connection destination management table

47] service entity management table

48 ⌋ connection management table

【図8】

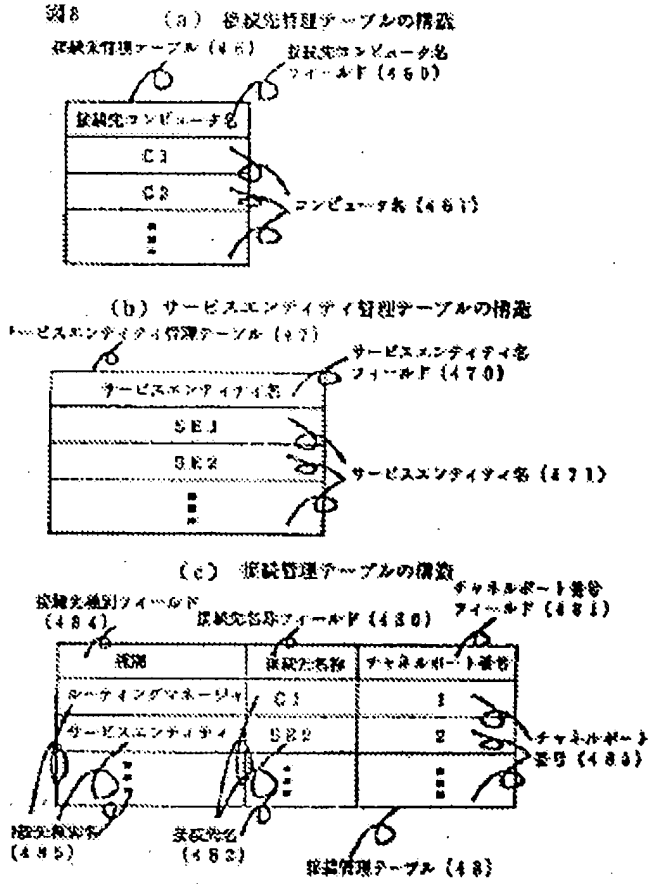


Fig. 8

(a) Structure of connection destination management table 46

460 ⌋ connection destination computer name field

461 ⌋ computer name

(table)

Connection destination computer name

C1
C2
⋮

(b) Structure of service entity management table

470 ⌋ service entity name field

471 ⌋ service entity name

Service entity name
SE1
SE2
⋮

(c) Structure of connection management table

48 ⌋ connection management table

480 ⌋ connection name field

481 ⌋ channel port number field

482 ⌋ connection destination name

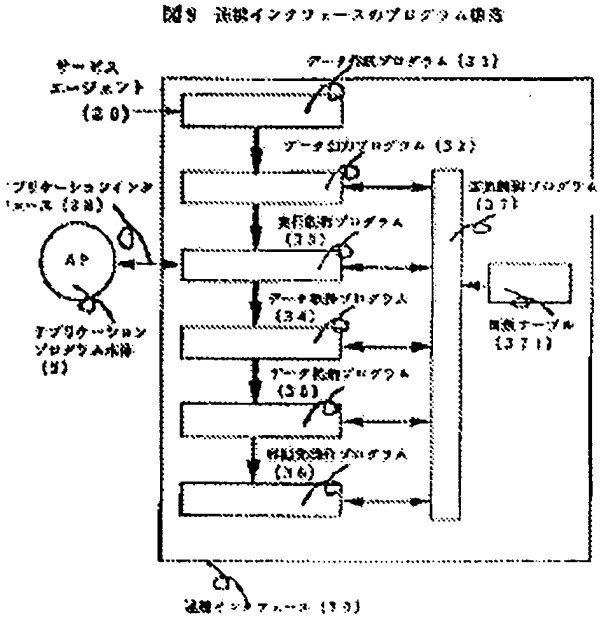
483 ⌋ channel port number

484 ⌋ connection destination kind field

485 ⌋ connection destination kind name

Kind	Connection destination name	Channel port No.
Routine manager	C1	1
Service entity	SE2	2
⋮	⋮	⋮

【図9】



3] application program body

20] service agent

30] cooperative interface

31] data preparation program

32] data output program

33] execution control program

34] data acquisition program

35] data storage program

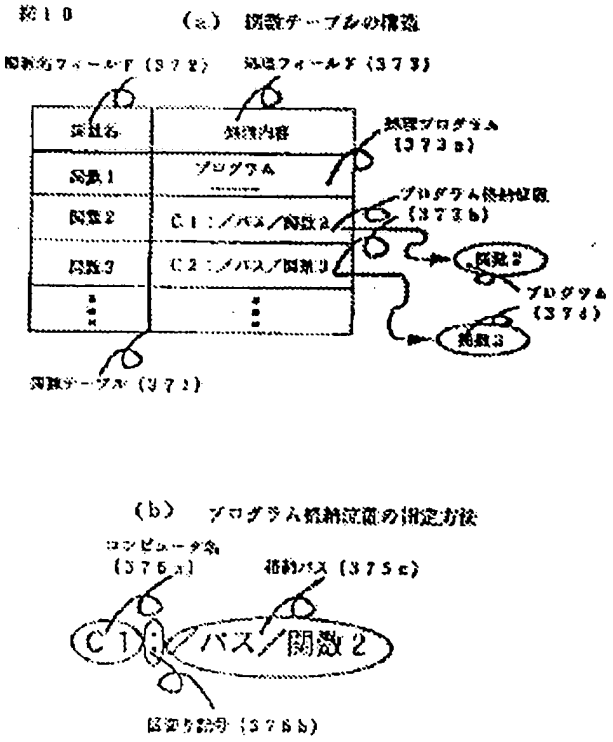
36] movement destination operation program

37] function control program

371] function table

38] application interface

【図１０】



(a)

(b) Structure of function table

371] function table

372] function name field

373 ⌋ processing field

373a ⌋ processing program

373b ⌋ program storage position

374 ⌋ program function 2 function 3

(table)

Function name	Processing contents
Function 1	Program ⌋⌋⌋⌋
Function 2	C1:/bus/function 2
Function 3	C2:/bus/function 3
⋮	⋮

(b) Method for assigning program storage position

C1:/pass/function 2

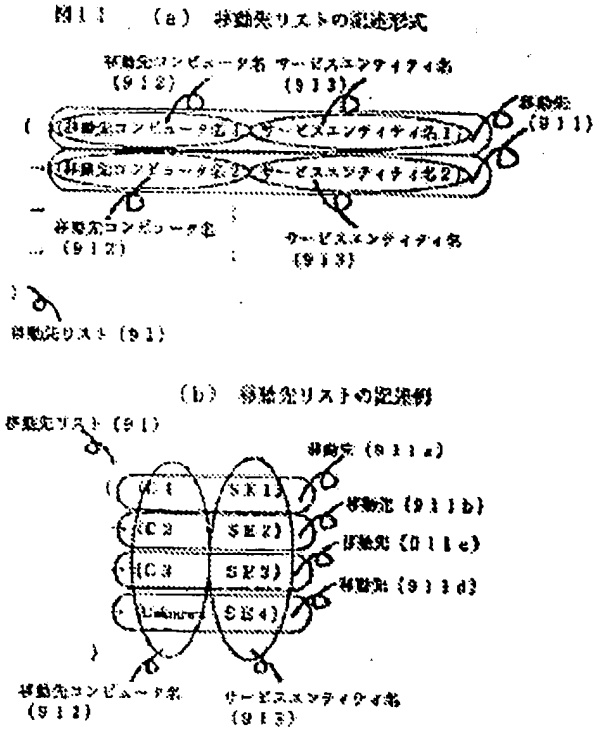
375a ⌋ computer name

375b ⌋ breakoff symbol

375c ⌋ storage pass

/24

【図１１】



(a) Description format of movement destination list

91 ⌋ movement destination list

911 ⌋ movement destination

912 ⌋ movement destination computer name

movement destination computer name 1

movement destination computer name 2

913 ⌋ service entity name

service entity name 1

service entity name 2

(b) Description example of movement destination list

91] movement destination list

911a] movement destination

911b] movement destination

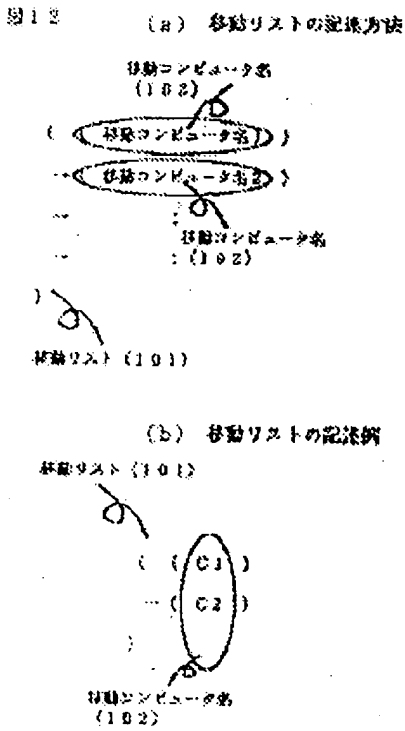
911c] movement destination

911d] movement destination

912] movement destination computer name

913] service entity name

【図12】



(a) Description format of movement list

101 ⌋ movement list

102 ⌋ movement computer name

movement computer name 1

movement computer name 2

(b) Description example of movement list

101 ⌋ movement list

102 ⌋ movement computer name

【図２１】

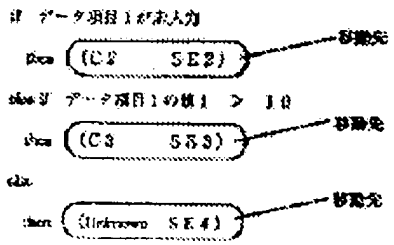


Fig. 21 Example of movement destination determining method based on combinations of data items

If data items are not inputted,

then (C2 SE2) ——— movement destination

else if the value of data item 1 > 10

then (C3 SE3) ———— movement destination

else

then (Unknown SE4) ———— movement destination

【図22】

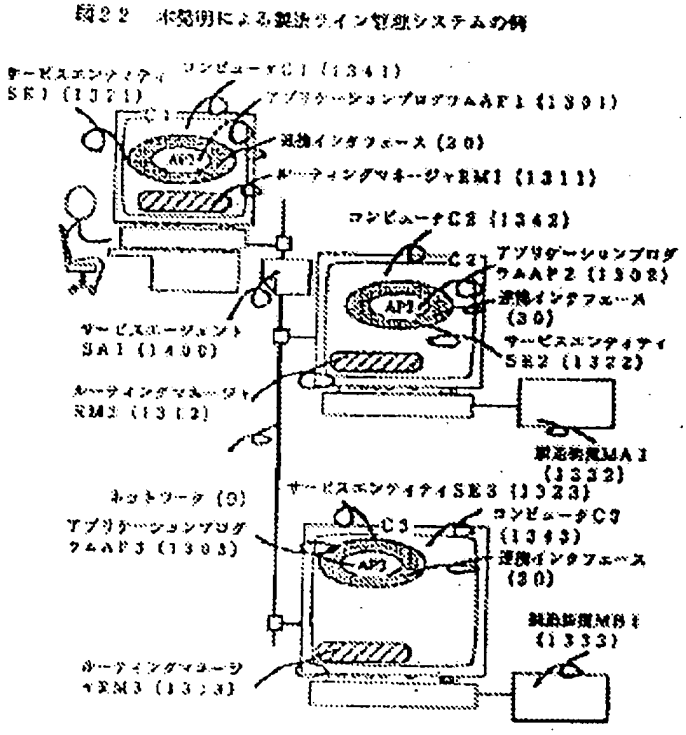


Fig. 22 Example of production line management system based on present invention

0 ⎫ network

30 ⎭ cooperative interface

1301 ⎫ application program AP1

1302 ⌋ application program AP2

1303 ⌋ application program AP3

1311 ⌋ routine manager RM1

1312 ⌋ routine manager RM2

1313 ⌋ routine manager RM3

1321 ⌋ service entity SE1

1322 ⌋ service entity SE2

1323 ⌋ service entity SE3

1332 ⌋ production machine MA1

1333 ⌋ production machine MB1

1341 ⌋ computer C1

1342 ⌋ computer C2

1343 ⌋ computer C3

1400 ⌋ service agent SA1

/25

【図１３】

図１３　サービスエージェントの移動処理手続き

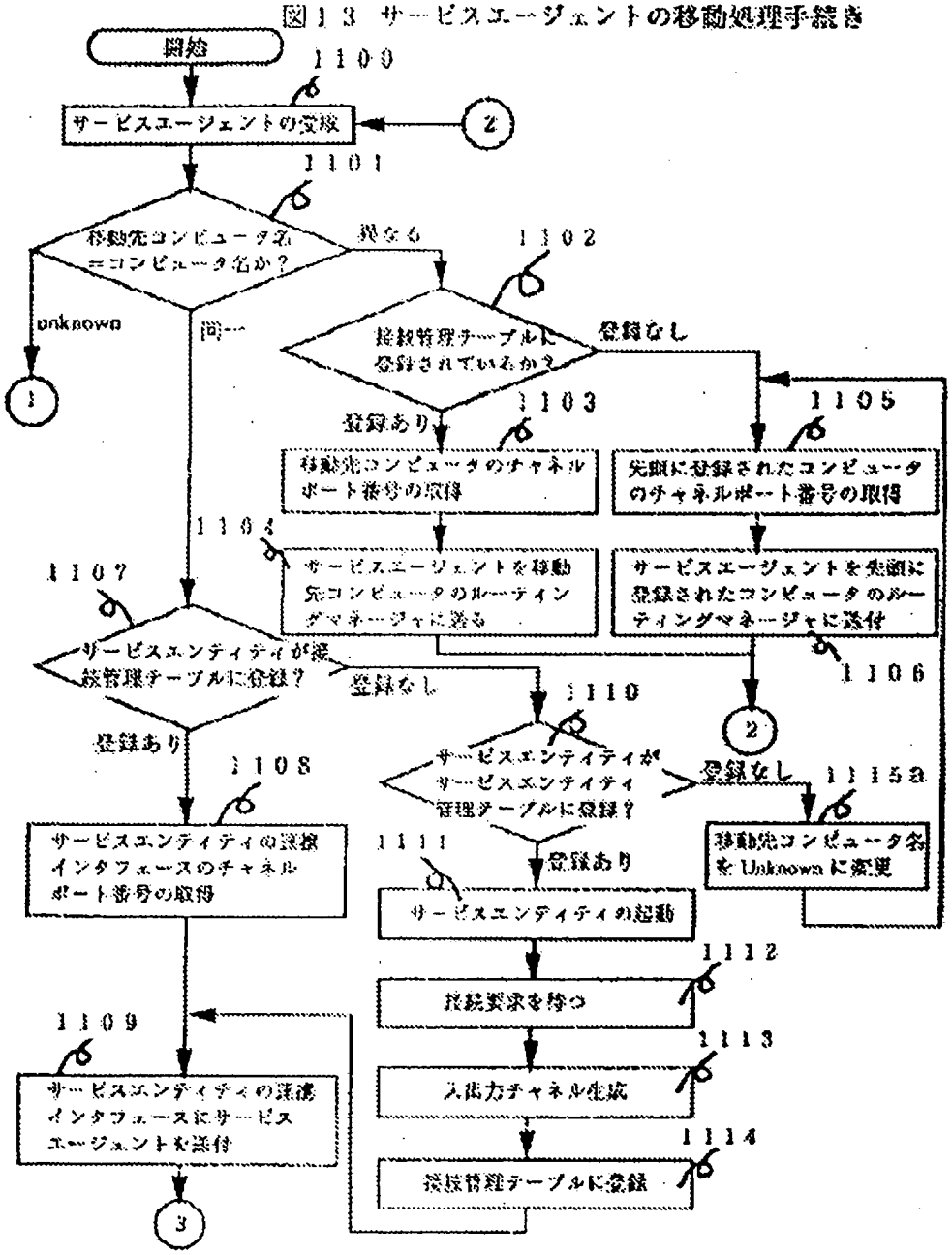


Fig. 13　Movement processing procedure of service agent

START

1100 ⌋ Receive service agent

1101 ⌋ destination computer name = computer name?

1102 ⌋ Is registered in connection management table?

1103 ⌋ Acquire channel port number of movement destination computer

1104 ⌋ Send service agent to routine manager of movement destination computer

1105 ⌋ Acquire channel port number of computer registered in head

1106 ⌋ Send service agent to routine manager of computer registered in head

1107 ⌋ Is service entity registered in connection management table?

1108 ⌋ Acquire channel port number of cooperative interface of service entity

1109 ⌋ Send service agent to cooperative interface of service entity

1110 ⌋ Is service entity registered in service entity management table?

1111 ⌋ Start service entity

1112 ⌋ Wait for connection request

1113 ⌋ Generate input/output channel

1114 ⌋ Register in connection management table

1115a ⌋ Change movement destination computer name to Unknown

(between 1101 and 1102) No

(between 1101 and 1107) Yes

(between 1102 and 1103) Yes

(between 1102 and 1105) No

(between 1107 and 1108) Yes

(between 1107 and 1110) No

(between 1110 and 1111) Yes

(between 1110 and 1115a) No

/26

【図14】

図14 サービスエージェントの移動処理手続き

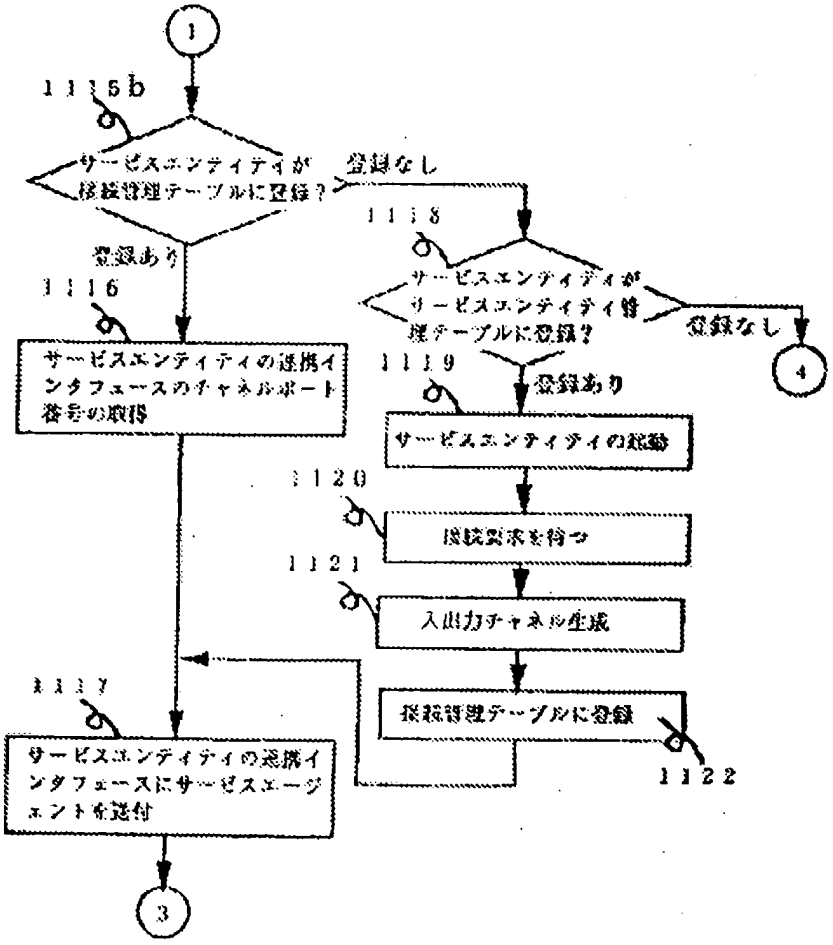


Fig. 14 Movement processing procedure of service agent

1115b ⌋ Is service entity registered in connection management table?

1116 ⌋ Acquire channel port number of cooperative interface of service entity

1117 ⌋ Send service agent to cooperative interface of service entity

1118 ⌋ Is service entity registered in service entity management table?

1119 ⌋ Start service entity

1120 ⌋ Wait for connection request

1121 ⌋ Generate input/output channel

1122 ⌋ Register in connection management table

(between 1115b and 1116) Yes

(between 1115b and 1118) No

(between 1118 and 1119) Yes

(between 1118 and 4) No

/27

[図15]

図15 サービスエージェントの移動処理手続き

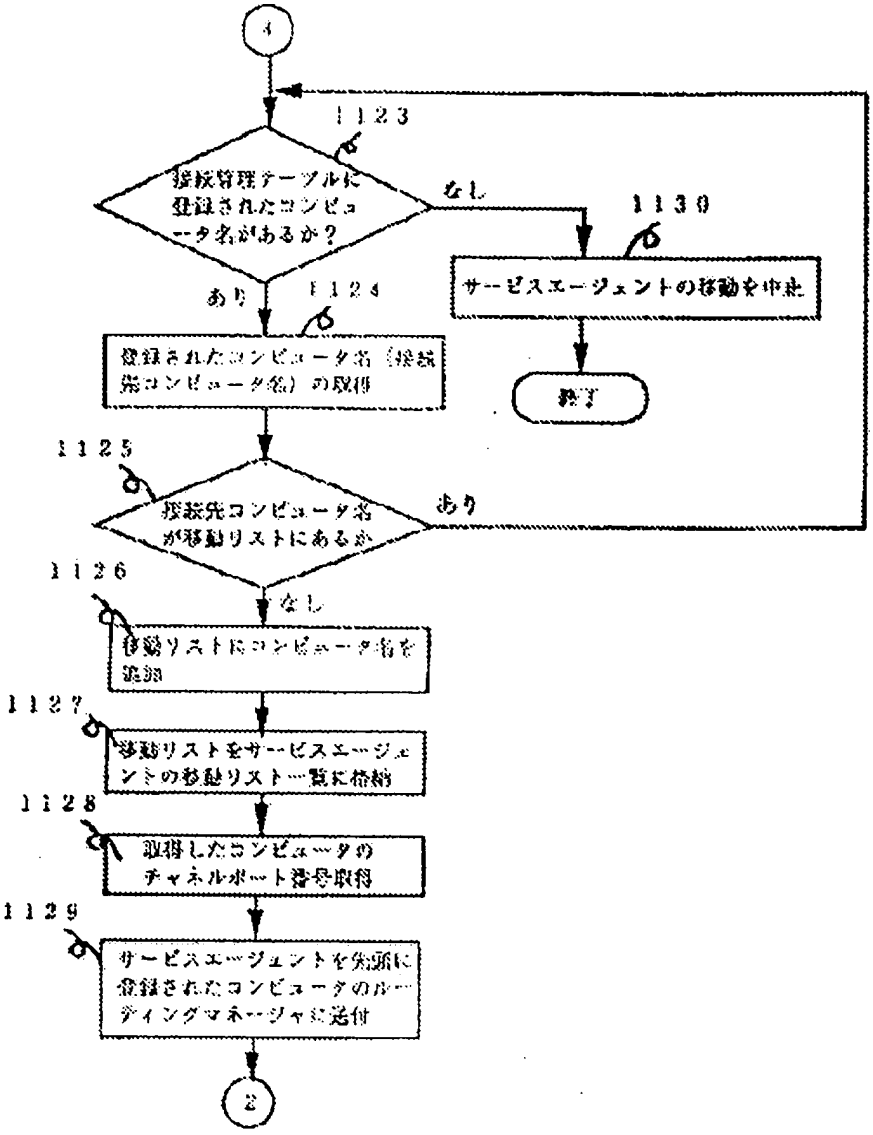


Fig. 15 Movement processing procedure of service agent

1123] Does registered computer name exist in connection management table?

1124] Acquire registered computer name (connection desti-

nation computer name)

1125 ⌋ Is connection destination computer name in movement list?

1126 ⌋ Supplement computer name in movement list

1127 ⌋ Store movement list in movement list summary of service agent

1128 ⌋ Acquire channel port number of acquired computer

1129 ⌋ Send service agent routine manager of computer registered in head

1130 ⌋ Stop movement of service agent

End

(between 1123 and 1124) Yes

(between 1123 and 1130) No

(between 1125 and 1123) Yes

(between 1125 and 1126) No

/28

【図16】

図16　サービスエージェントの移動処理手続き

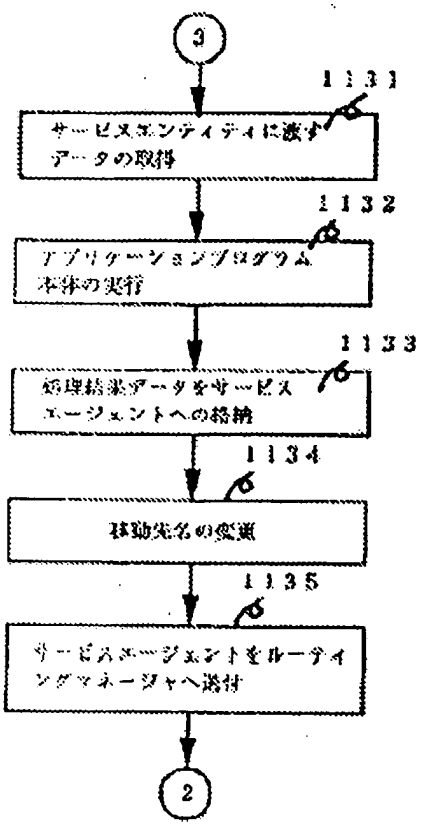


Fig. 16　Movement processing procedure of service agent

1131 ⎫　Acquire data transferred to service entity

1132 ⎫　Execute application program body

1133 ⎫　Store processing result data into service agent

1131 ⎫　Change movement destination name

1135 ⎫　Send service agent to routine manager

【図２３】

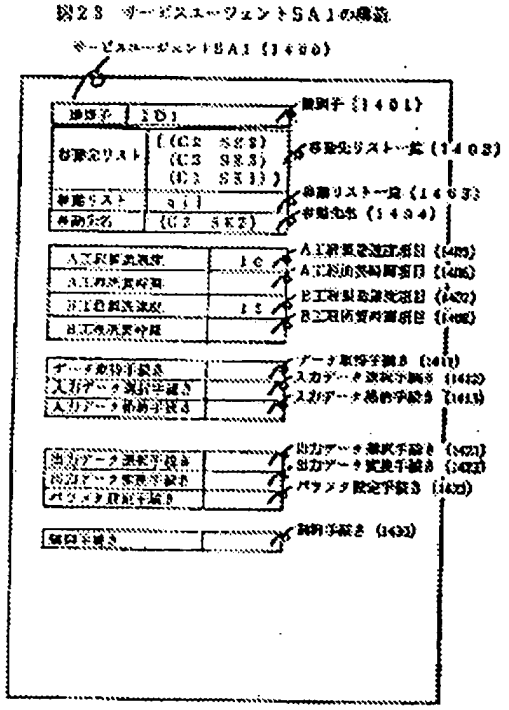


Fig. 23 Structure of service agent SA1

1400) service agent SA1

1401) identifier

1402) movement destination list summary

1403) movement list summary

1404) movement destination name

1405) process A manufacturing speed item

1406) process A required time item

1407) process B manufacturing speed item

1408 ⌋ process B required time item

1411 ⌋ data acquisition procedure

1412 ⌋ input data selection procedure

1413 ⌋ input data storage procedure

1421 ⌋ output data selection procedure

1422 ⌋ output data conversion procedure

1423 ⌋ parameter setup procedure

1430 ⌋ control procedure

(table)

Identifier	ID1
Movement destination list	((C2 SE2) (C3 SE3) (C1 SE1))
Movement list	nil
Movement destination name	(C2 SE2)

Process A manufacturing speed	10
Process A required time	
Process B manufacturing speed	15
Process B required time	

Data acquisition procedure	
Input data selection procedure	
Input data storage procedure	

Output data selection procedure	
Output data conversion procedure	
Parameter setup procedure	

Control procedure	

/29

【図17】

図17　連携インタフェースの処理フロー

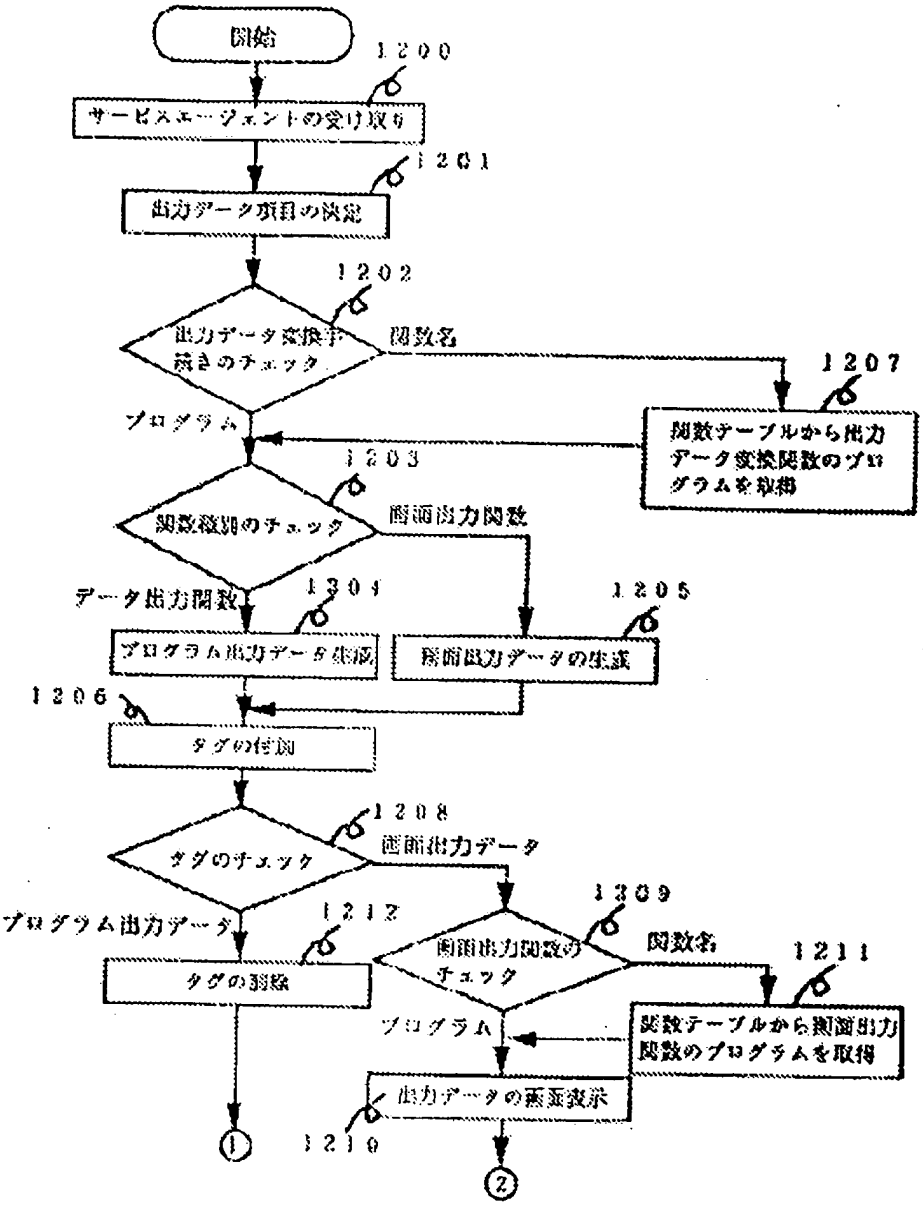


Fig. 17　Processing flow of cooperative interface

Start

1200 ⌋　Receive service agent

1201) Determine output data item

1202) Check output data conversion procedure

1203) Check kind of function

1204) Generate program output data

1205) Generate picture output data

1206) Attach tag

1207) Acquire program of output data conversion function from function table

1208) Check tag

1209) Check picture output function

1210) Display picture of output data

1211) Acquire program of output data function from function table

(between 1202 and 1203) Program

(between 1202 and 1207) Function name

(between 1203 and 1204) Data output function

(between 1203 and 1205) Picture output function

(between 1208 and 1209) Picture output data

(between 1208 and 1212) Program output data

(between 1209 and 1210) Program

(between 1209 and 1211) Function name

/30

【図18】

図18　連携インタフェースの処理フロー

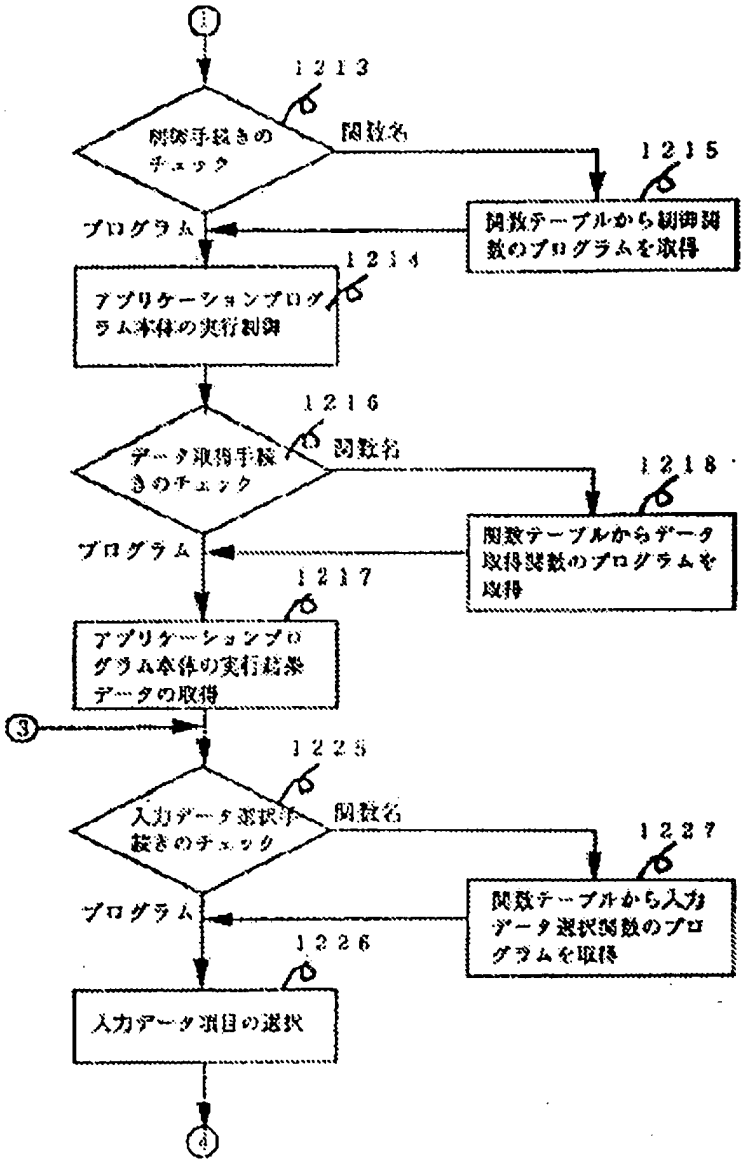


Fig. 18　Processing flow of cooperative interface

1213 ｝　Check control procedure

1214 ｝　Control execution of application program body

1215 ⌋ Acquire program of control function from function table

1216 ⌋ Check data acquisition procedure

1217 ⌋ Acquire execution result data of application program body

1218 ⌋ Acquire program of data acquisition function from function table

1225 ⌋ Check input data selection procedure

1226 ⌋ Select input data item

1227 ⌋ Acquire program of input data selection function from function table

(between 1213 and 1214) Program

(between 1213 and 1215) Function name

(between 1225 and 1226) Program

(between 1225 and 1227) Function name

/31

【図19】

図19　連携インタフェースの処理フロー

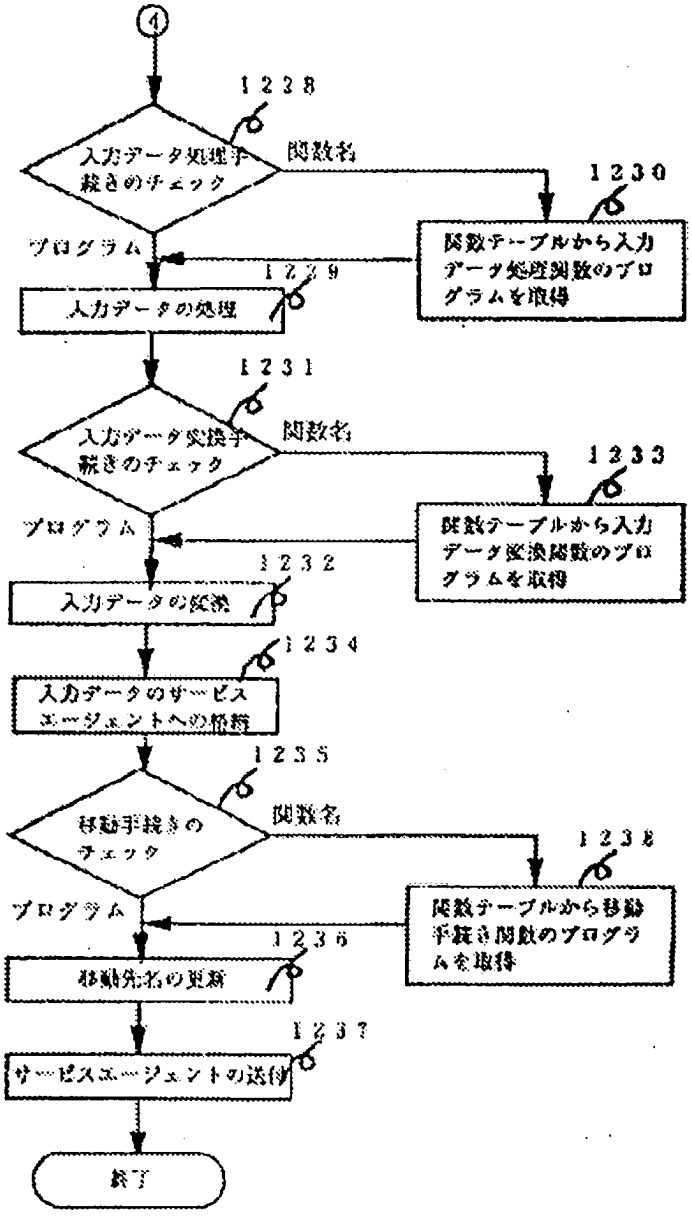


Fig. 19　Processing flow of cooperative interface

1228 ⌋　Check input data processing procedure

1229 ⌋ Processing of input data

1230 ⌋ Acquire program of input data processing function from function table

1231 ⌋ Check input data conversion procedure

1232 ⌋ Convert input data

1233 ⌋ Acquire program of input data conversion function from function table

1234 ⌋ Store input data into service agent

1235 ⌋ Check movement procedure

1236 ⌋ Renew movement destination name

1237 ⌋ Send service agent

1238⌋ Acquire program of movement procedure from function table

End

(between 1228 and 1229) Program

(between 1228 and 1229) Function name

(between 1231 and 1232) Program

(between 1231 and 1233) Function name

(between 1235 and 1236) Program

(between 1235 and 1238) Function name

[図20]

図２０　連携インタフェースの処理フロー

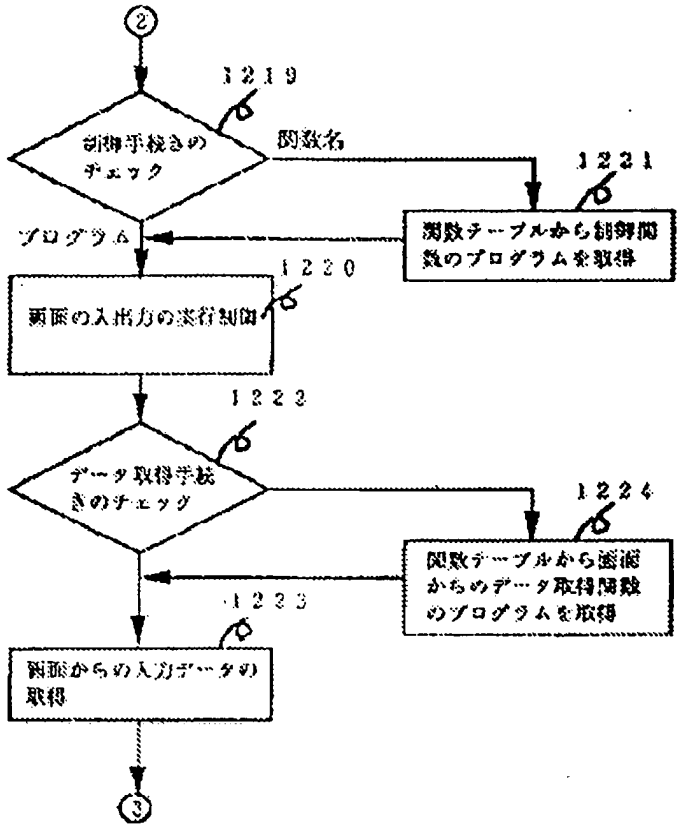


Fig. 20　Processing flow of cooperative interface

1219 ｝　Check control procedure

1220 ｝　Control execution of picture input/output

1221 ｝　Acquire program of control function from function table

1222 ｝　Check data acquisition procedure

1223 ｝　Acquire input data from picture

1224) Acquire program of data acquisition from picture from function table

(between 1219 and 1220) Program

(between 1219 and 1221) Function name